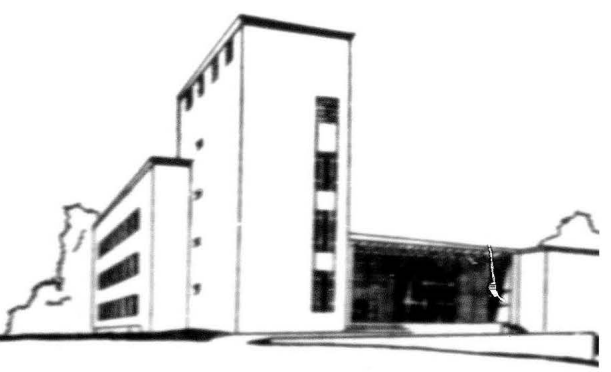
Carnegie Institute of Technology

Pittsburgh 13, Pennsylvania



GRADUATE SCHOOL of INDUSTRIAL ADMINISTRATION

William Larimer Mellon, Founder

O.N.R. Research Memorandum No. 117

PROBABILISTIC AND PARAMETRIC LEARNING COMBINATIONS OF LOCAL JOB SHOP SCHEDULING RULES

by

Wallace B. Crowston
Fred Glover
Gerald L. Thompson
Jack D. Trawick

November, 1963

Graduate School of Industrial Administration

Carnegie Institute of Technology

Pittsburgh, Pennsylvania 15213

This paper was written as part of the contract "Planning and Control of Industrial Operations," with the Office of Naval Research, at the Graduate School of Industrial Administration, Carnegie Institute of Technology. Reproduction of this paper in whole or in part is permitted for any purpose of the United States Government. Contract Nonr-760 (01), Project NR-047011.

TABLE OF CONTENTS

TABLE OF CONTENTS

PREFACE

During the academic year 1962-63, the authors conducted an extensive series of tests on methods for solving job shop scheduling problems. Reference [5] gives some good background material, and Chapters 3 and 12-18 are especially relevant to the present study. Most of the methods studied here stem from the probabilistic combination of local job shop scheduling rules proposed by Fischer and Thompson in [1]. In addition, other methods such as direct search [3] and permutation techniques (recommended in [6]) have also been tested and are reported on. The exact methods of complete enumeration and integer programming were not tried here, but see [2] and [4]. The contribution of F. Glover to this paper lies in the parametric method of Chapter VI.

The one firm conclusion of our studies is that probabilistic and parametric combinations of local rules offer significant improvements over other methods of searching for good or optimal schedules. Moreover, they yield results in a reasonable amount of computer time, and hence are susceptible of immediate application. We have not compared these two methods as to the amount of time they will take, and hence cannot, at present, choose between probabilistic and parametric methods. However, we feel that the parametric method deserves considerable further study. The permutation approach advocated by Sherman and Reiter seems to be very slow, but does offer the advantage that a probability estimate can be given as to the worth of the results obtained. However, it seems to us to be computationally impractical for present computer speeds and costs. Similarly, direct search, as we have used it, seems not to be competitive.

CHAPTER I

PROBABILISTIC COMBINATIONS OF LOCAL JOB SHOP SCHEDULING RULES

1. INTRODUCTION

This chapter describes a computer program for combining in a "probabilistic" way six local job shop scheduling rules. The rules included in this program are the shortest imminent operation (SIO), longest remaining time (LRT), first-in, first-out (FIFO), and machine slack (MS) rules. Also, a fill-in rule, which does not enter the decision process, is used which modifies the straight forward application of the rules. The modified rules differ from the standard rules in that an operation is not scheduled if a good of higher priority and presently being processed on another facility will arrive prior to the expected completion of the highest priority operation in the queue. If this situation occure, the facility is held idle until the new good arrives. The FIFO rule, however, does not need such a look-ahead feature since goods are processed on a first-come, first-serve basis. An operation is not delayed because of the possible arrival of a higher priority operation which is now standing idle in some other queue. There is one exception to the above "one-step" look-ahead which will be discussed under the description of the machine slack rule in (6) below.

The fill-in rule attempts to fill in idle gaps resulting from the above look-ahead feature by specifying that whenever an idle delay is incurred the entire queue shall be scanned for any operation that can be scheduled to fill the gap without delaying the start of the operation which is being awaited. This is done by scanning the current SIO list in reverse order and selecting those goods (if any) whose time (or summed times) is not greater than the idle gap.

(1) The SIO (shortest imminent operation) rule says that whenever a facility becomes available, select that good in the queue which has the shortest machining time on the facility.

(2) The LRT (longest remaining time) rule says select that good in the queue which has the most machining time remaining until completion.

(3) The JS (job slack per operation remaining) rule says select that good in the queue which has the least job slack per operation remaining until completion. Job slack is determined for each good by subtracting remaining machining time from an arbitrarily established finish or due date.

(4) The LIO (longest imminent operation) rule says select that good in the queue which has the longest machining time on the given facility.

(5) The FIFO (first-in, first-out) rule says select that good which arrived first in the queue.

(6) The MS (machine slack) rule says select that good in the queue which has the least machining time remaining until it arrives at the facility with the longest remaining machining time. If all goods have already been processed on the most critical facility (i.e., the most remaining machining time), then the earliest arrival at the next most critical facility is used as a criteria for selection (and so on).

Two versions of the rule were tested. In the first, the "one-step" look-ahead feature, as previously described, was used. This feature only considered those goods which were currently being processed on another facility whose next operation would be on the available facility. In the second, a "two-step" look-ahead feature was used which also considered those goods which were currently being processed on another facility and which would arrive at the available facility after the next operation.

2. DESCRIPTION OF THE PROGRAM

A. Initializing

In the initialization segment of the program, the following list structures are set up.

1. Scheduling Rule List:

One list is set up for each rule used in the program. Each of these lists contains all the goods that are available for scheduling on each facility at any given time. The goods in each facility's queue are ordered according to their priority under the respective rules with the highest priority good at the head of the list. Each list contains identical goods at all times, but the ordering of the goods under each of the rules is not (necessarily) the same.

2. Facility Clock List:

The facility clock list designates the earliest time at which each facility is available for scheduling an operation.

3. Current Operation Counter List:

This list designates the current operation being performed (or to be performed when the appropriate facility becomes available) on each good.

4. Decision Point Counter List:

This list designates the next decision point in the decision vector to be used for each facility when a rule choice is to be made.

5. Machine Priority List:

This list has all facilities ordered according to total remaining machining time with the facility having the greatest total time at the top of the list.

6. Decision Vectors:

Each facility has a decision vector consisting of the same number of elements as there are operations to be performed on the facility. Each element consists of a string of 3 digit numbers, each of which represents the cumulative probability of selecting one of the rules lower in the list of rules than the particular rule to which the numbers relate. The element to be referred to when a decision on a particular facility is to be made is determined from the decision point counter list. Because of the fill-in feature, it may not be necessary to refer to the decision vector for each good scheduled on a given facility and the number of elements used in the facility's decision vector may be less than the number of operations.

7. List of Fill-in Goods:

This list designates those goods which have been determined as being satisfactory for filling idle gaps on facilities resulting from the look-ahead previously discussed. If an idle gap occurs on a given facility and only one filler good can be used, it is immediately scheduled and no goods are stored in the facility's waiting filler list. If, however, there is more than one filler good, one is immediately scheduled and the remaining goods are temporarily stored in the waiting list to be scheduled in order as the facility becomes available.

8. Schedule:

Both a temporary working schedule and a best achieved schedule are maintained. The schedules show the start and finish times of each operation for each good on each machine. From the best achieved schedule, a schedule can be outputted in form equivalent to a Gantt Chart.

B. Scheduling

Scheduling consists of the following primary operations.

(1) Determine if facility (i = k, 2,...,n) is available for scheduling a good at time t = 1, 2,..., completion of scheduling.

a. If not, i ← i + 1 and go to 1. If no facilities are available, increase time period (t ← t + 1) being examined by one.

b. If yes, go to 2.

(2) Determine the good number just completed on the given facility.

a. Increase the completed good's current operation counter by one.

b. Remove the completed good from all scheduling rule lists for the given facility. If all operations have been completed on the good, go to 3.

c. Place the completed good in its next machine queue. Determine the priority of the good under each of the scheduling rules and place in the appropriate position in each list.

(3) Check for waiting fill-in goods in given facility's list of fill-in goods. If there is a waiting fill-in good, remove the good from the list and go to 9. If there is no waiting fill-in good, go to 4.

(4) Check to see if the facility is being held idle awaiting the arrival of a higher priority good.

a. If yes, i ← i + 1 and go to 1.

b. If no, go to 5.

(5) Determine if there is a good currently in the given facility's queue.

a. If yes, go to 6.

b. If no, i ← i + 1 and go to 1.

(6) Select one of the decision rules.

(7) Determine the good with the highest priority under the selected decision rule.

a. If the FIFO rule is selected, go to 9.

b. Determine the finish time of the good if scheduled on the given facility.

c. Determine if any good currently being processed on another facility, whose next operation is on the given facility, will arrive before the time determined in (b) and will have a higher priority under the decision rule selected than the good selected in (7). If machine slack is the selected decision rule, either a "one-step" or a "two-step" look-ahead is used.

1. If there is such a good, go to 8.

2. If there is not, go to 9.

(8) Determine the arrival time of the good for which the facility will be held idle.

a. Determine the idle gap which will occur.

b. Check for filler goods.

1. If there are no filler goods, $i \leftarrow i + 1$ and go to 1.

2. If there are filler goods, place the first on the working schedule noting start and finish times for the appropriate good operation number and facility. Place all remaining filler goods on the fill-in waiting list for the facility. $i \leftarrow i + 1$ and go to 1.

(9) Place the good selected on the working schedule noting starting time and finish time for the appropriate good operation number and facility. Update the Machine Availability Clock List by noting the finish time of the good just scheduled plus one unit of time.

a. If all goods are scheduled, go to 10.

b. If not, i ← i + 1 and go to 1.

(10) Evaluate current schedule relative to best schedule to date.

a. If better, store the best achieved schedule.

b. If more runs are to be made, go to the initialization part of the program to reset all list structures and then repeat steps 1-9.

3. DISCUSSION OF THE PROGRAM

Of the steps outlined, only steps 6 and 8b require additional clarification.

Step 6 calls for selecting one of the six decision rules included in the program. To do this, reference is made to the appropriate element in the facility's decision vector as determined from the facility's decision point counter. The elements are used successively as each new decision is encountered. Each element consists of 5 three-digit numbers. Each of the three-digit numbers represents the cumulative probability of selecting one of the rules lower in the list of rules than the particular rule to which the numbers relate. In this program, the rules are ordered in the following way:

1. Shortest imminent operation
2. Longest remaining time
3. Machine slack
4. Job slack per operation remaining
5. Longest imminent operation
6. First-in, first-out

The order in which the rules appear is arbitrary, but some time saving can be achieved by having those rules with the highest probability of being selected at the head of the list. Since six rules were used in the program, it was convenient to represent 100% as 600.

As an example of the use of the decision vector element, consider the case where all rules are to have an equiprobable chance of selection. The decision vector element would be 500400300200100. Only 5 three-digit numbers are necessary since the sixth rule will automatically be selected if none of the first five are. By comparing a pseudo-random, 3-digit number in the range 1-600 with each of the 3-digit numbers in the decision vector element, it can be determined which rule is to be selected. A number of 359, for example, would specify the selection of the machine slack rule.

To check for filler goods when an idle gap occurs on a facility as a result of awaiting the arrival of a higher priority good, the following procedure is followed. All goods in the facility's queue are examined by scanning the SIO rule list in reverse order (i.e., from longest to shortest required machining time) to determine if their required machining time is not greater than the idle gap. From among the set of all goods (if any) that satisfy this condition, that good whose machining time (or those goods whose summed machining times) most completely fills the gap is (are) specified as filler goods.

4. DESCRIPTION OF TEST RESULTS

Only the 20X5X5 problem (as given in [1]) has been used thus far in testing the 6-rule program. Since no learning process had yet been built into the program, the various combinations of rules tested were initially determined by intuitive guesses as to what might generate good schedules, and later, search was directed toward the region of combinations which appeared to result in the best schedules.

The scheduling times achieved under various combinations and probabilities of the rules are plotted in Charts 1-9. In the majority of test runs, the machine slack rule (either 1-step or 2-step look-ahead) was

applied 100% of the time in the initial decision points. It was found that there was some "best" number of decision points through which to apply the 100% machine slack rule followed by some combination of other rules (see Chart 4 for the 1-step look-ahead case and Chart 8 for the 2-step look-ahead case). The best SIO and LRT combination probabilities were found to be in the vicinity of 70% and 30% respectively. Also, it appears (with the limited runs available) that only slight moves of 10% away from these probabilities results in a greater change in schedule times achieved than was the case for the 6×6 problem.

Although the mean of all schedules and the best schedule generated using 100% machine slack (2-step look-ahead) during the first 10 decision points followed by 70% SIO and 30% LRT was much better than those generated by applying the same through a lesser number of decision points (see Chart 8), it is believed that this also might reduce the number of alternative sequences of goods which might achieve the minimum possible schedule.

Chart 1

Time Path - Total Schedule Time

20 × 5 Problem

Non-Learning

(30 Schedules)

SIO-LRT Equiprobable

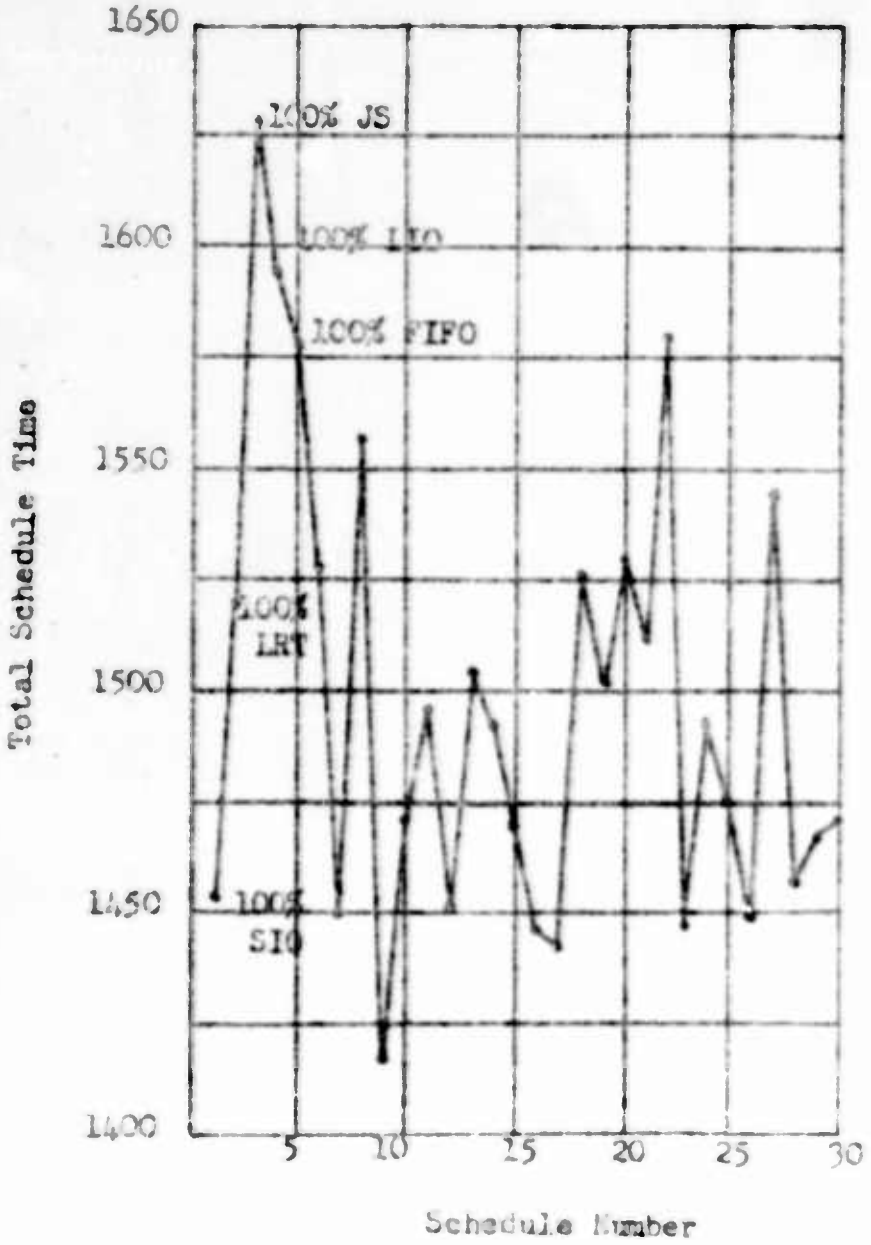


Chart 2

Time Path - Total Schedule Time

20 × 5 Problem

Non-Learning

(50 Schedules)

100% Machine Slack First 5 Decision Points

(1 step look ahead)

60% SIO 40% LRT Remainder

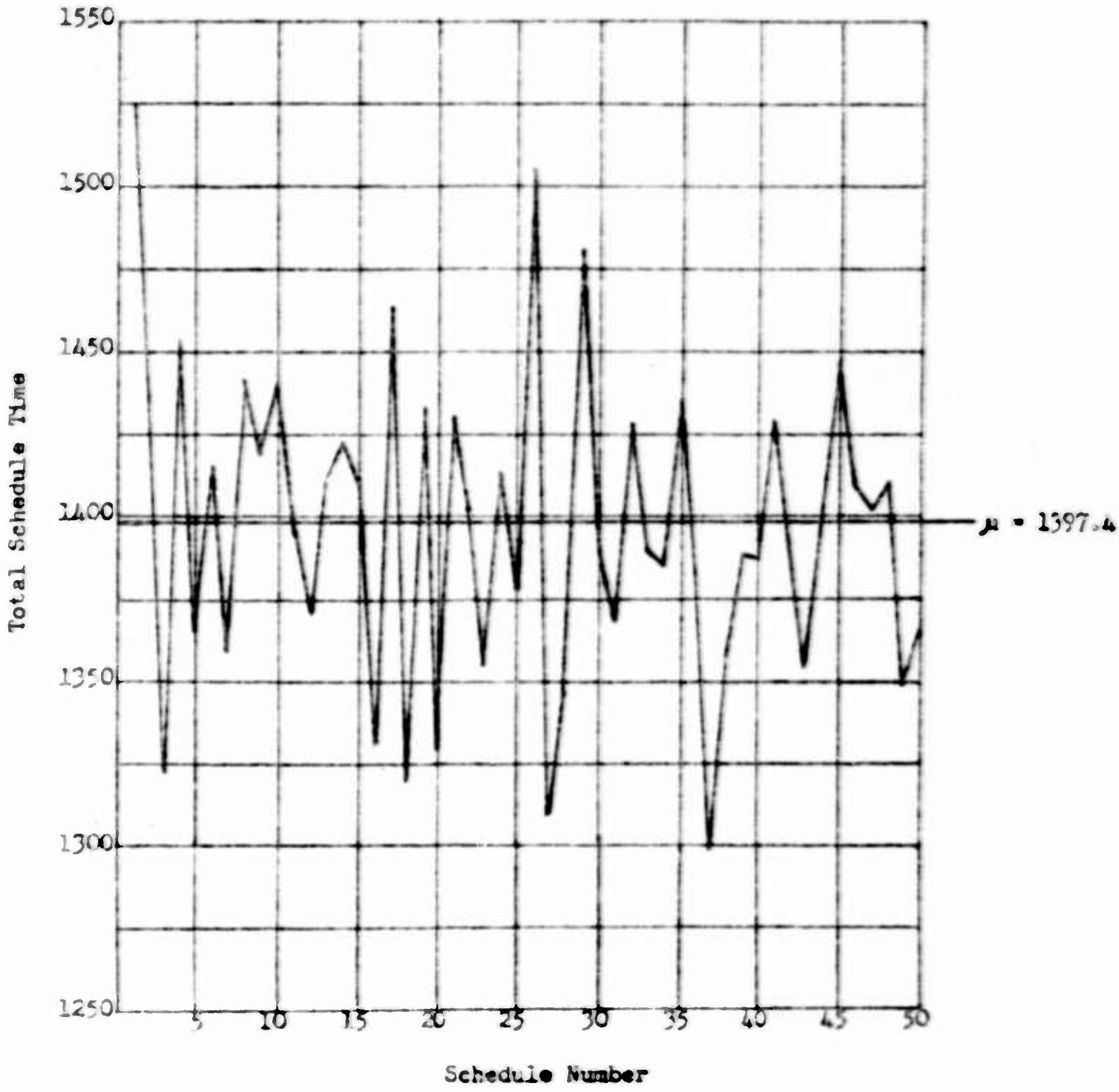


Chart 3

Time Path - Total Schedule Time

20 x 5 Problem

Non-Learning

(50 Schedules)

100% Machine Slack First 5 Decision Points

(1 Step Look Ahead)

75% SIO 25% LRT Remainder

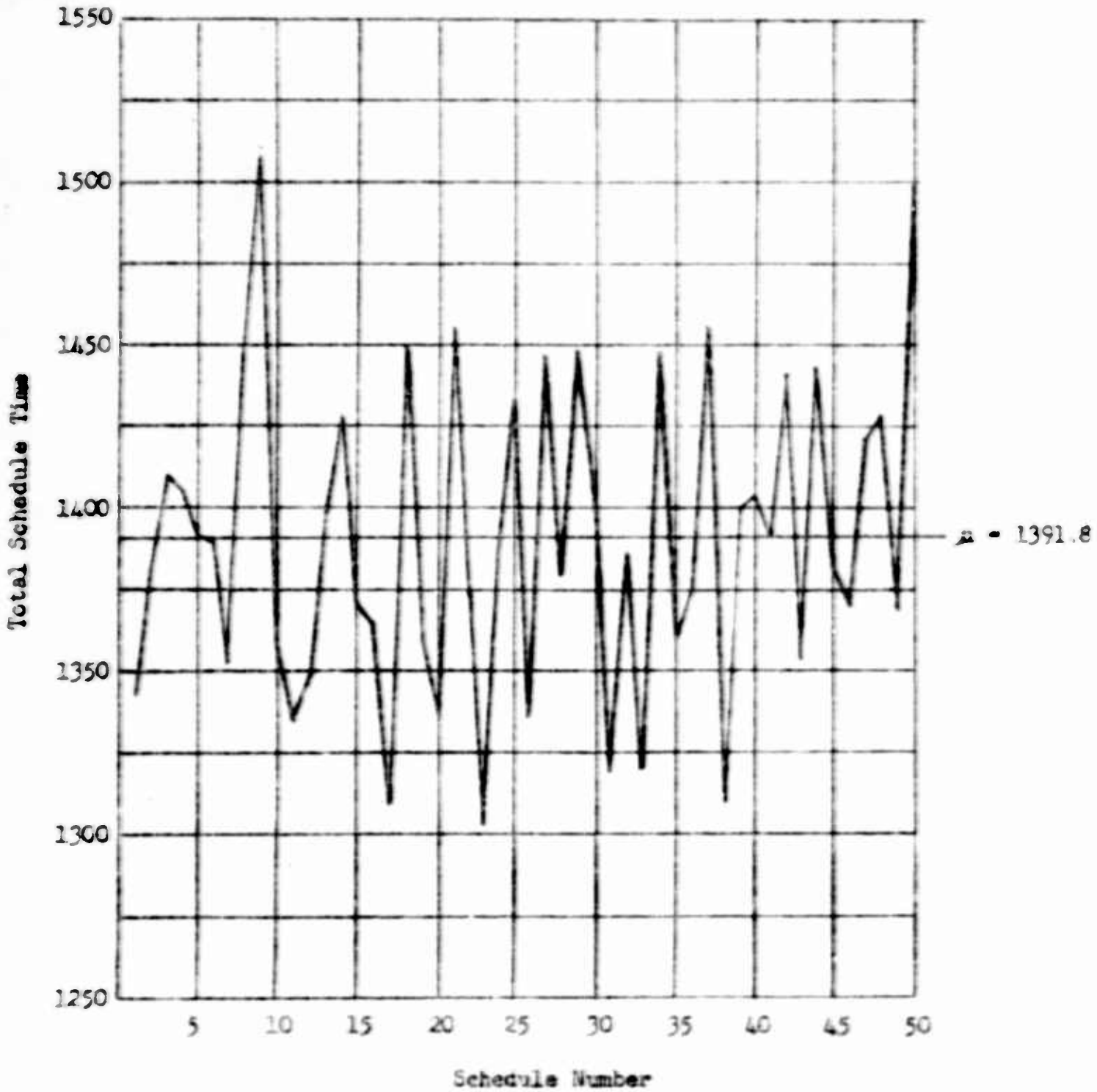
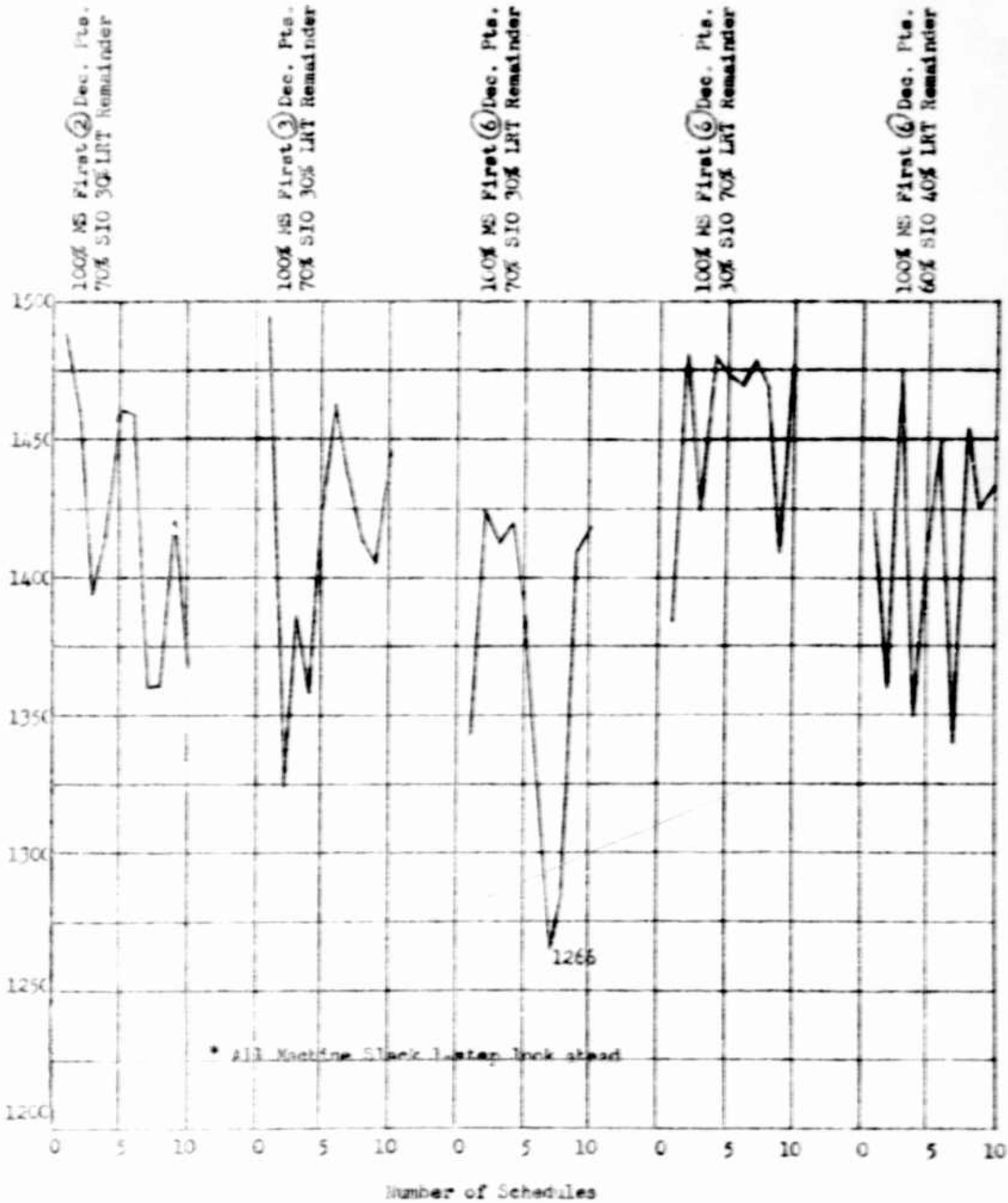


Chart 4

Time Path - Total Schedule Time 20 × 5 Problem

Non-Learning (Various Combinations)



* All Machine Slack Lag-step look ahead

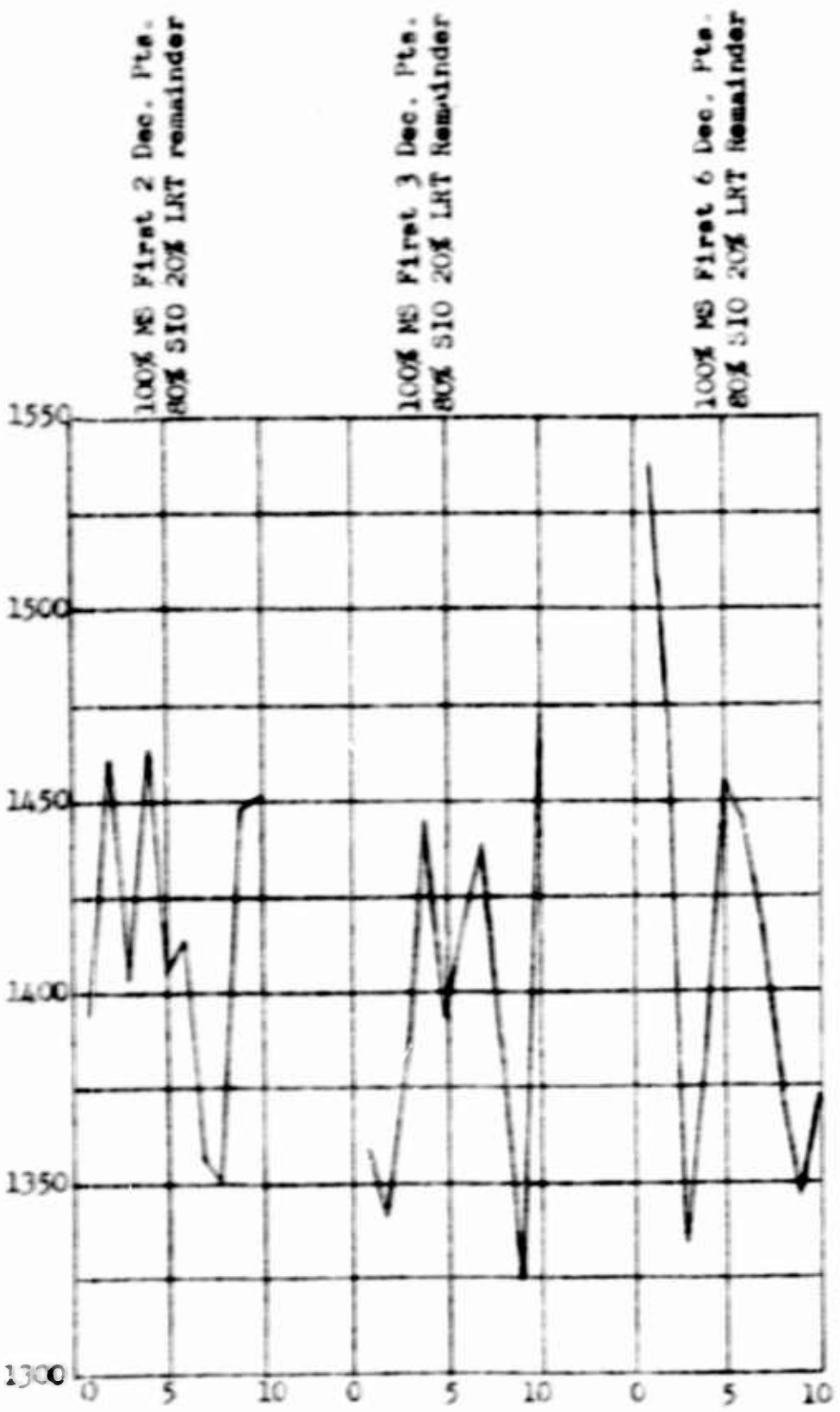
Chart 5

Time Path - Total Schedule Time

20 × 5 Problem

Non-learning

(Various Combinations)



• All Machine Slack 1-step look ahead

Chart 6

Time Path - Total Schedule Time

20 • 5 Problem Non-Learning

(Various combinations)

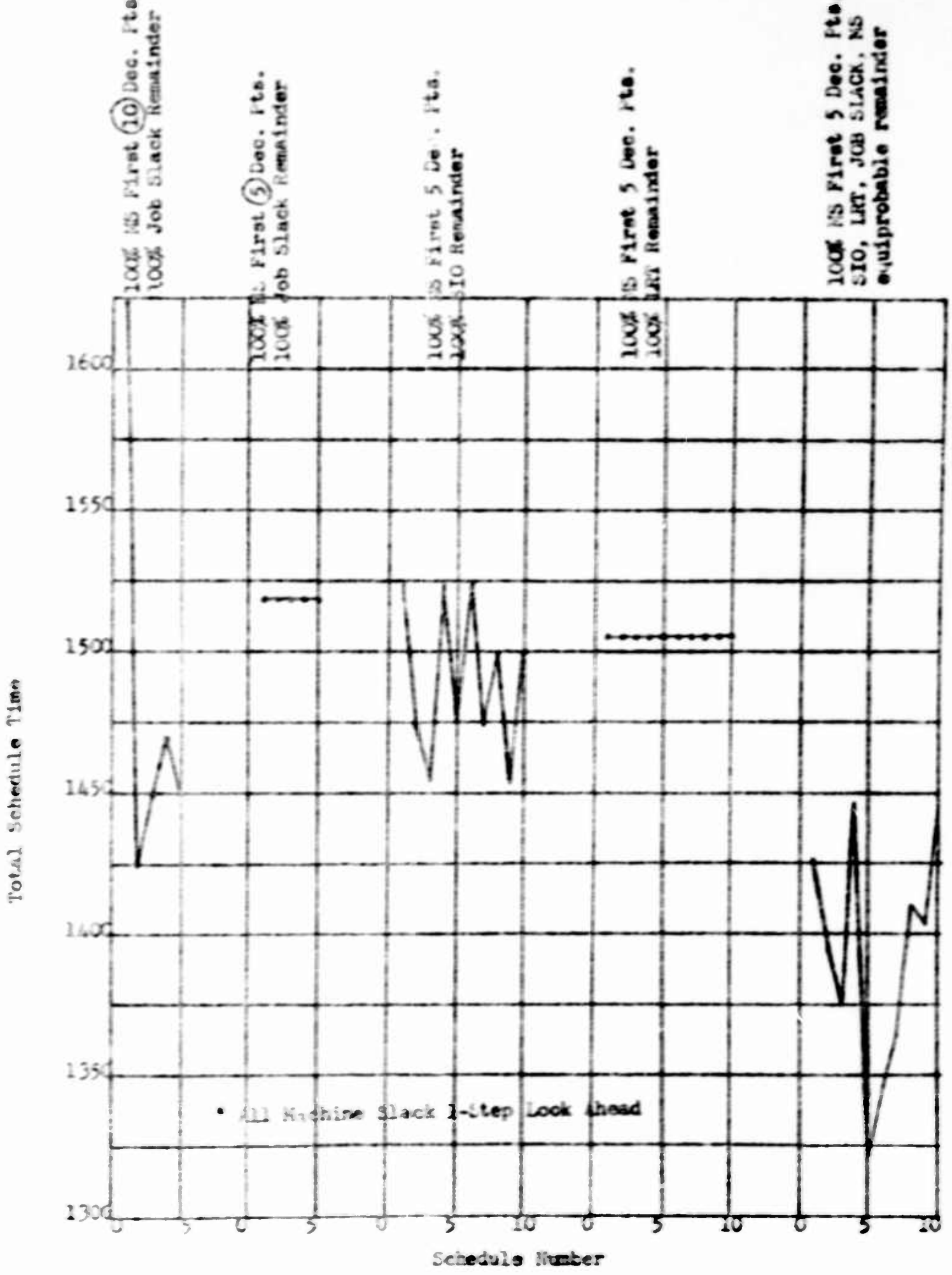


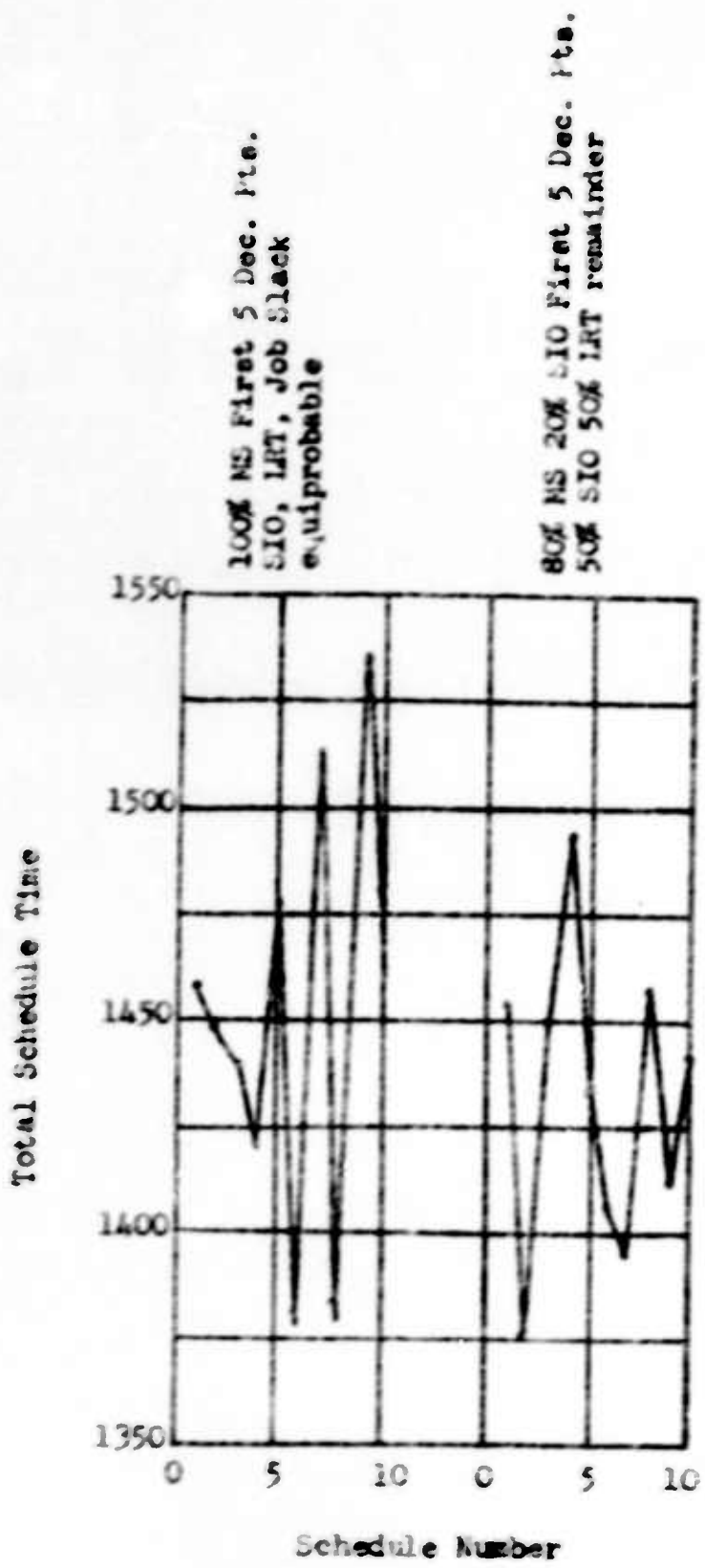
Chart 7

Time Path - Total Schedule Time

20 × 5 Problem

Non-Learning

(Various Combinations)



* All machine slack 1-step look ahead

Chart 8

Time Path-Total Schedule Time

20 × 5 Problem

Non-Learning

All MS 2-Step Look-Ahead

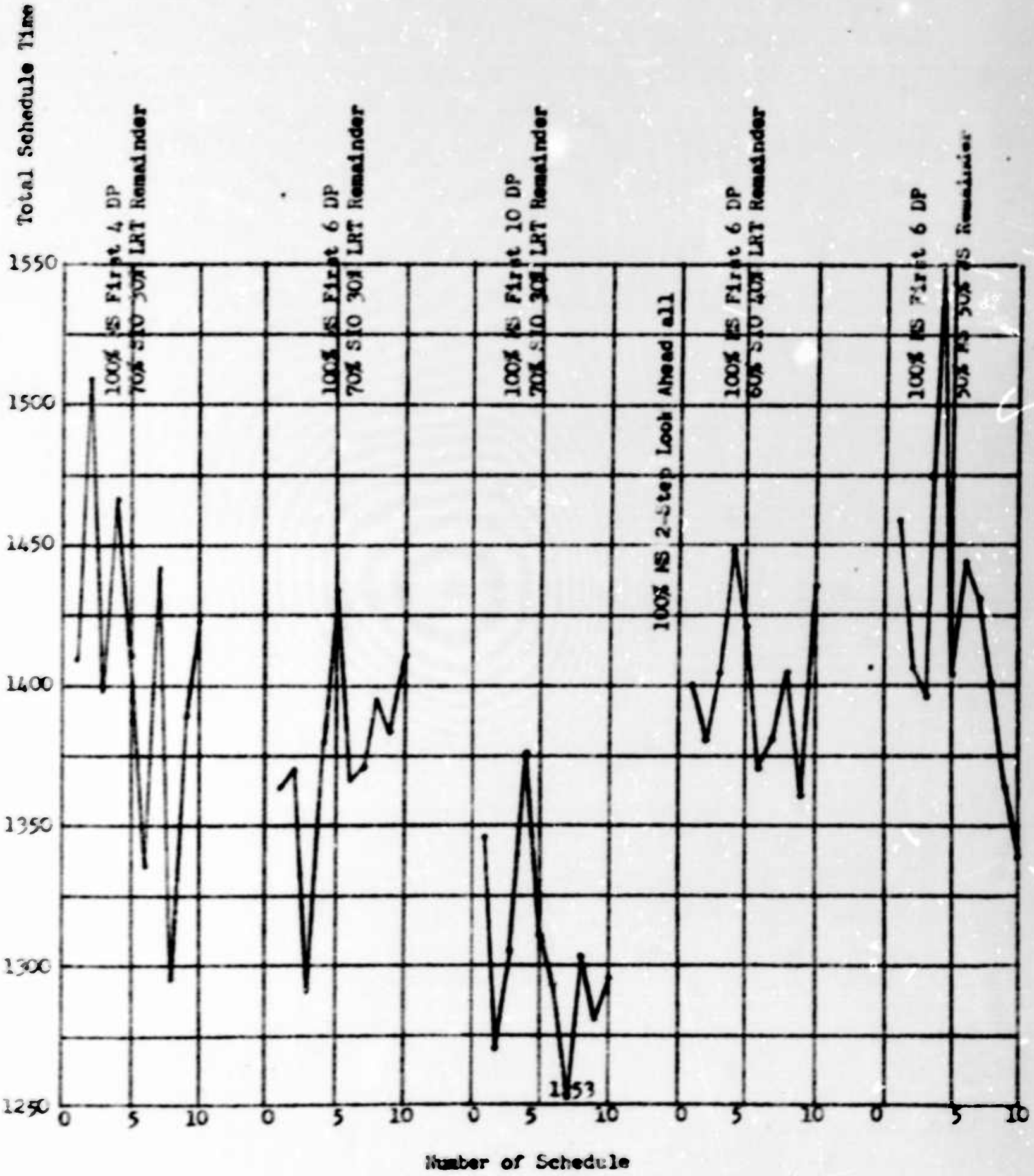


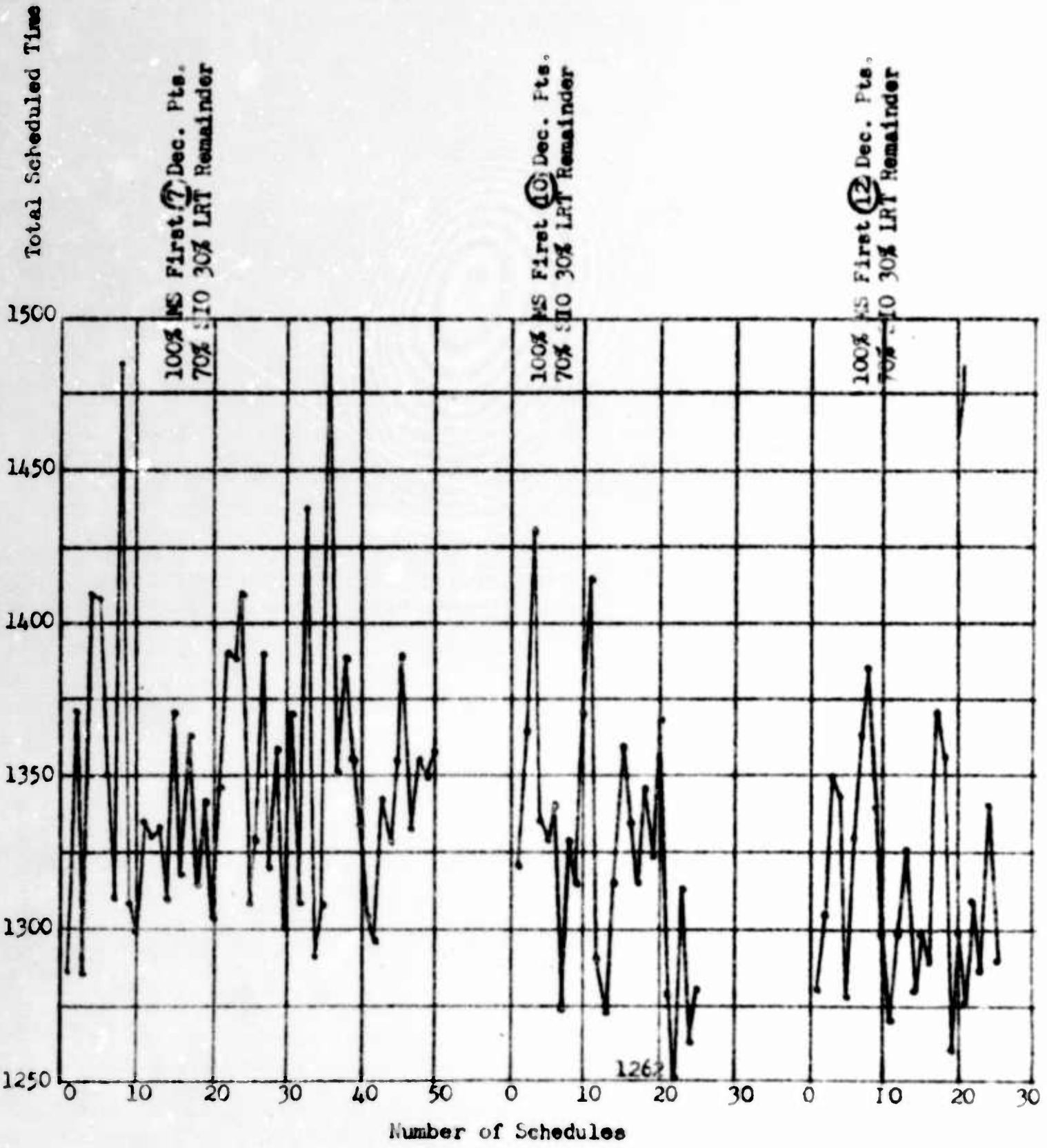
Chart 9

Time Path - Total Schedule Time

20 × 5 Problem

Non-Learning

* All Machine Slack 2-Step Look Ahead



CHAPTER II

TEST RESULTS FROM THE JOB SHOP SCHEDULING PROGRAM

1. INTRODUCTION

In the Job Shop Scheduling program described in Chapter I, any probabilistic combination of six scheduling rules can be used to generate a schedule. These rules are SIO, LRT, LIFO, FIFO, Job Slack and Machine Slack. This program is described in detail elsewhere. In the following series of tests only two of the rules, SIO and LRT, are used and they are applied only to 6X6 matrices. The questions examined are as follows.

1. What is the effect of various probabilistic combinations of SIO and LRT rules on the mean and the variance of a series of generated schedules? Do some ratios produce a higher number of optimum schedules?

2. Does the range of machine times in a specific problem have an effect on the difficulty of obtaining an optimum schedule? Does it affect the mean or variance of a series of schedules?

3. What is the effect of increasing the number of schedules in a series on the mean, the variance of the series and on the number of optimum schedules generated?

4. What is the result of changing the probabilities of the LRT and SIO rules within a given schedule?

2. THE METHOD

Data for answering questions 1, 2 and 3 was gathered in the following way. Four scheduling problems were randomly generated, each having six goods to be processed on six machines. In the first of these, the range of processing times on each machine was from one to ten days. Processing times in the second problem were generated between two and nine days, inclusive. Similarly, the times of the third and fourth problems varied from

three to eight days, and from four to seven days respectively. In these four problems the order of the operations for each good was held constant so that a comparison of the results would be more meaningful. These problems are shown in Figures 1-4 along with the means of the machine processing time and the lowest schedule time generated throughout the series of tests. This will be referred to as the "optimum schedule."

For each of the four problems two hundred schedules were generated for each of eleven combinations of SIO and LRT probabilities. These combinations began at 0% SIO, 100% LRT and changed in ten steps of 10% to 100% SIO, 0% LRT. In total then forty-four series of two hundred schedules each were generated. For each series of two hundred schedules, the mean, the variance and the number of optimum schedules were recorded. The above procedure was then repeated with each sequence reduced to ten schedules. The results of all the tests may be found in the graphs of Figures 7-18.

Data for question four was gathered in the following manner. In the determination of a schedule for a 6X6 problem, thirty-six scheduling decisions must be made. Four series of the 10:1 ratio problem were generated. In series 1 the first eighteen decisions were made using the SIO rule, and the last eighteen were made using the LRT rule. In series two the order was reversed with the LRT rule used for the first half. In series three for each schedule percentage SIO used was decreased gradually throughout the scheduling. For the first four decision points the ratio of SIO to LRT was 90:10. For the next four decision points the ratio was 80:20. This continued throughout the scheduling so that for the last four decisions the ratio was 10:90, SIO:LRT. In the fourth sequence the process was repeated, but with decreasing LRT. The results are shown in Figure 19.

Scheduling Problems

Problem 1

Machining Time Ratio 10:1

Good 1		Good 2		Good 3		Good 4		Good 5		Good 6	
Machine	Days	Machine	Days	Machine	Days	Machine	Days	Machine	Days	Machine	Days
3	1	2	8	3	5	2	5	3	9	2	3
1	3	3	5	4	4	1	5	2	3	4	3
2	6	5	10	6	8	3	5	5	5	6	9
4	7	6	10	1	9	4	3	6	4	1	10
6	3	1	10	2	1	5	8	1	3	5	4
5	6	4	4	5	7	6	9	4	1	3	1

Mean Machining Time - 4.5 days

Optimum Schedule 55 days

Figure 1

Scheduling Problems

Problem 2

Machining Time Ratio 9:2

Good 1		Good 2		Good 3		Good 4		Good 5		Good 6	
Machine	Days	Machine	Days	Machine	Days	Machine	Days	Machine	Days	Machine	Days
3	6	2	9	3	6	2	2	3	6	2	7
1	9	3	2	4	6	1	3	2	3	4	2
2	9	5	6	6	5	3	7	5	3	6	2
4	6	6	2	1	8	4	8	6	4	1	9
6	5	1	5	2	3	5	4	1	4	5	3
5	3	4	7	5	8	6	8	4	5	3	9

Mean Machining Time - 5.4 days

Optimum Schedule 50 days

Figure 2

Scheduling Problem

Problem 3

Machining Time Ratio 8:3

Good 1		Good 2		Good 3		Good 4		Good 5		Good 6	
Mach- ine	Days	Mach- ine	Days	Mach- ine	Days	Mach- ine	Days	Mach- ine	Days	Mach- ine	Days
3	8	2	8	3	6	2	8	3	8	2	3
1	5	3	7	4	8	1	8	2	4	4	8
2	4	5	6	6	6	3	3	5	3	6	5
4	5	6	3	1	8	4	6	6	6	1	4
6	7	1	6	2	3	5	5	1	3	5	5
5	8	4	7	5	6	6	5	4	3	3	3

Mean Machining Time • 5.6 days

Optimum Schedule 57 days

Figure 3

Scheduling Problem

Problem 4

Machining Time Ratio 7:4

Good 1		Good 2		Good 3		Good 4		Good 5		Good 6	
Mach-ine	Days	Mach-ine	Days	Mach-ine	Days	Mach-ine	Days	Mach-ine	Days	Mach-ine	Days
3	6	2	7	3	6	2	6	3	6	2	6
1	7	3	7	4	6	1	6	2	7	4	4
2	4	5	4	6	5	3	7	5	7	6	7
4	7	6	6	1	5	4	5	6	5	1	7
6	5	1	7	2	6	5	6	6	1	5	6
5	4	4	4	5	7	6	4	4	7	3	6

Mean Machining Time • 6 days

Optimum Schedule 55 days

Figure 4

Total Machining Time For Each Good

Problem	Good 1	Good 2	Good 3	Good 4	Good 5	Good 6
10:1	26	47	34	35	25	30
9:2	38	31	36	32	25	32
8:3	37	37	37	35	27	28
7:4	33	35	35	34	38	36

Figure 5

Total Operating Time For Each Machine

Problem	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5	Machine 6
10:1	40	26	26	22	40	43
9:2	38	33	36	34	27	26
8:2	34	30	35	37	33	32
7:4	36	36	38	33	34	32

Figure 6

Figure 7

Relation of Mean of Schedules to % SIC, LRT

Job ratio 10:1

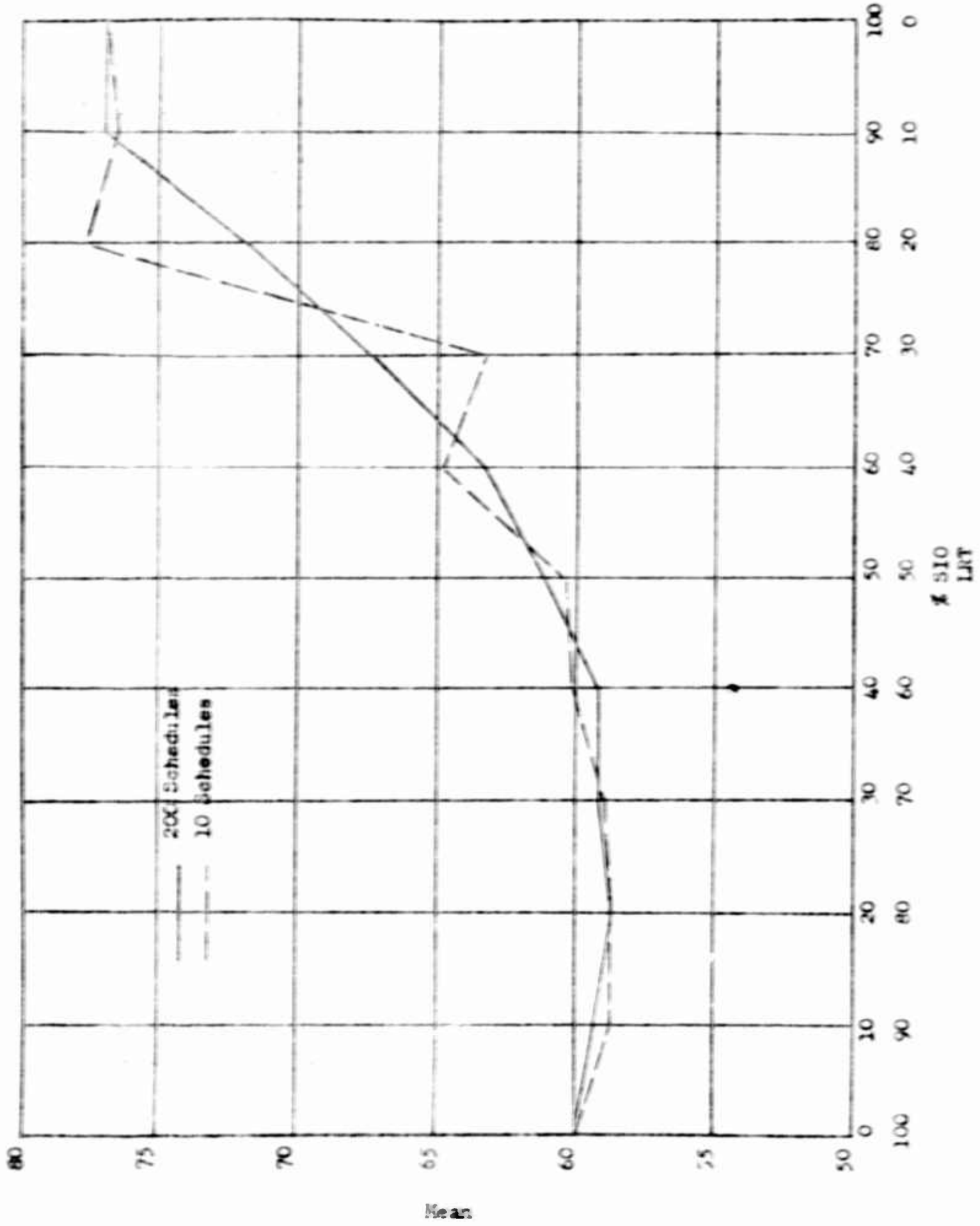


Figure 8

Relation of Mean of Schedules to % SIC, LRT

Job ratio 9:2

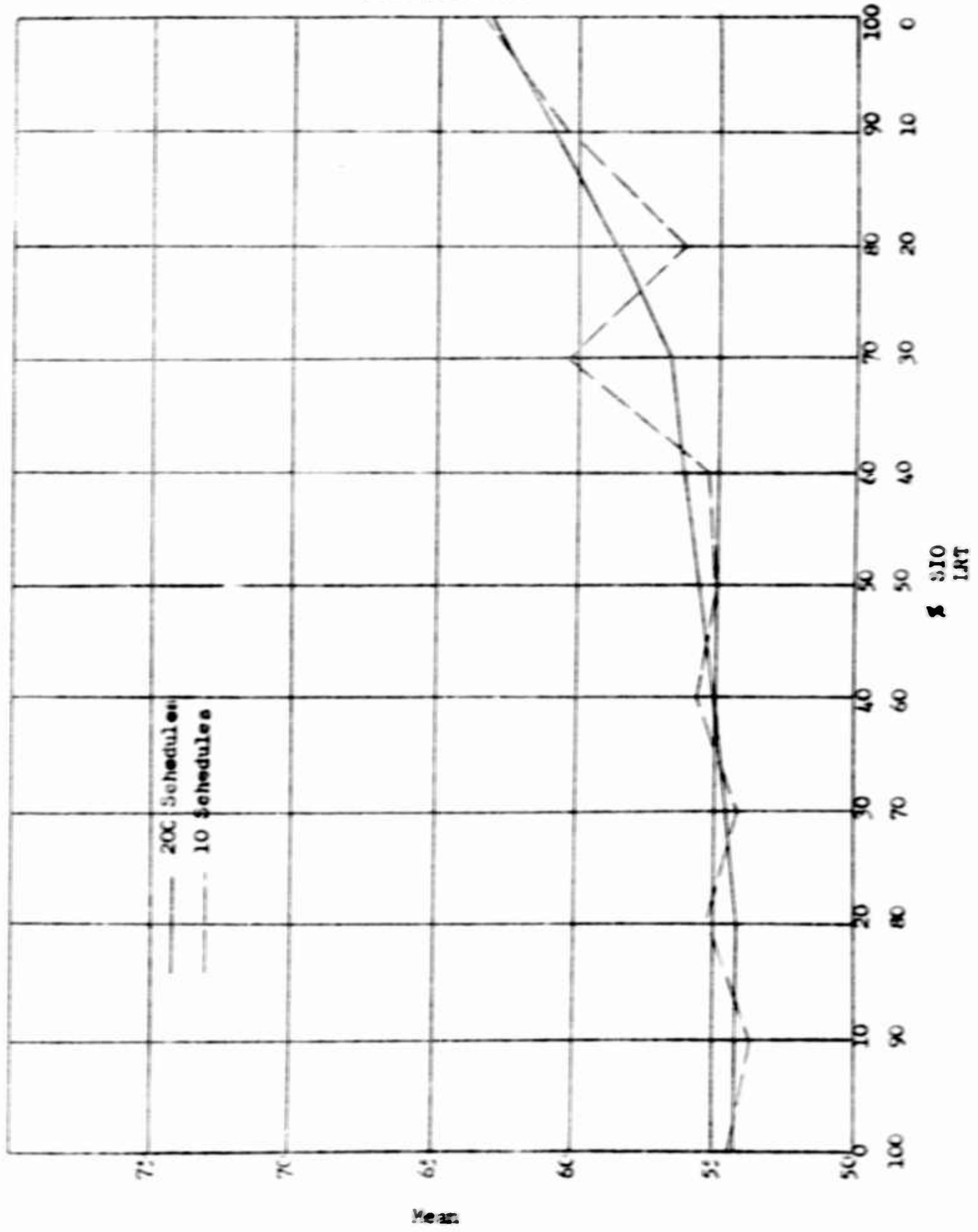


Figure 9

Relation of Mean of Schedules to % SIO, LRT

Job ratio 8:3

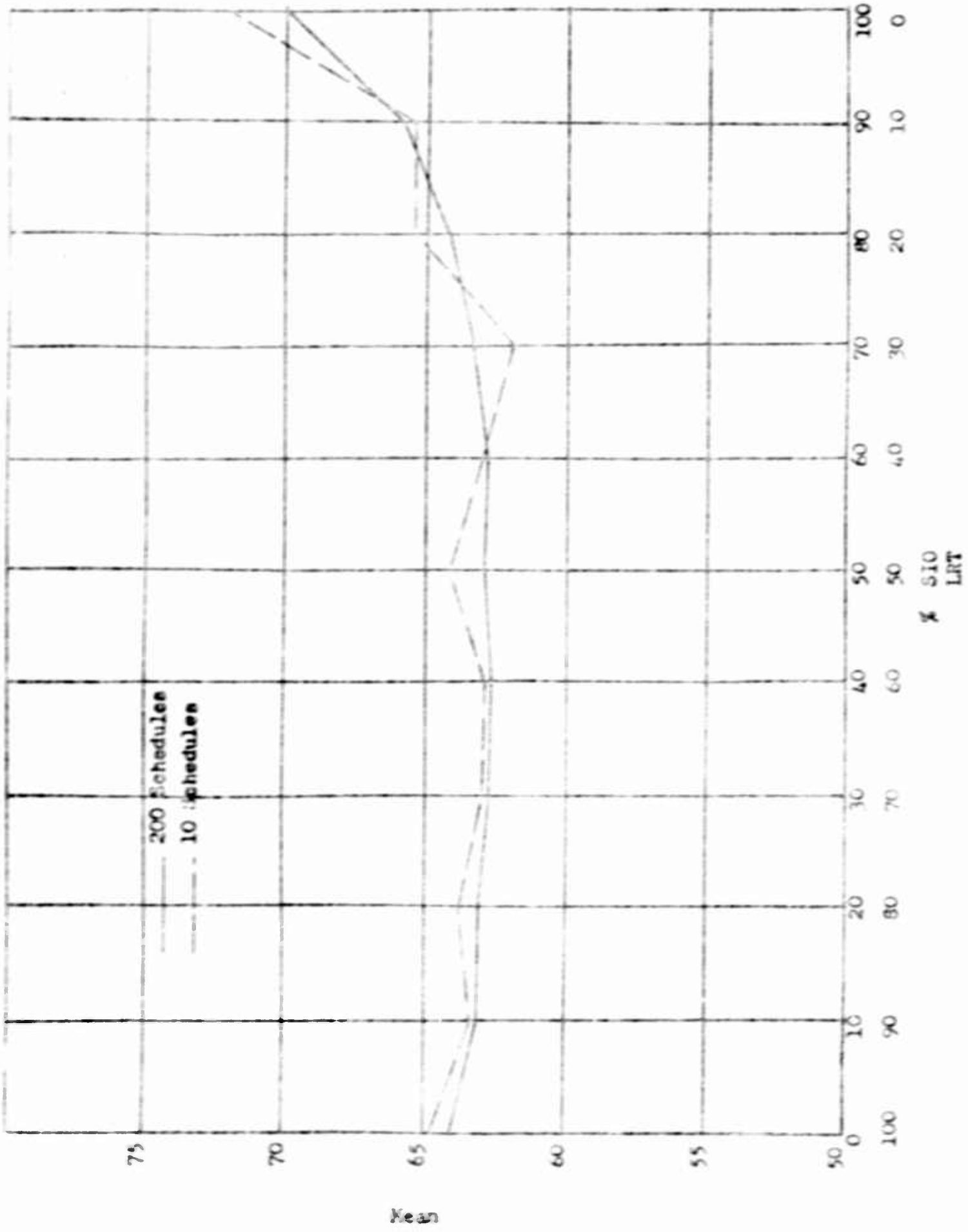


Figure 10

Relation of Mean of Schedules to % SIO, LRT

Job ratio 7:4

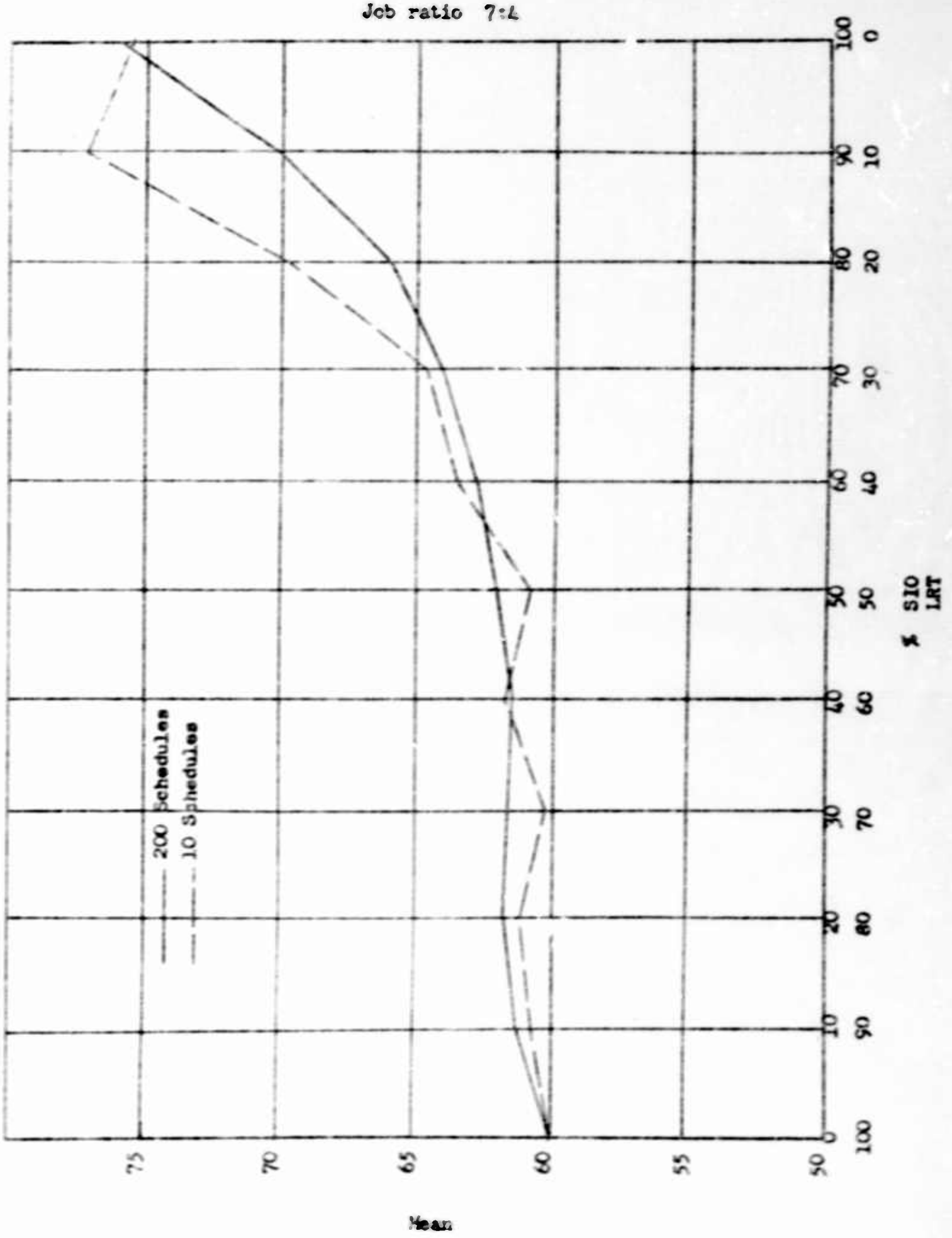


Figure 11

Relation of Mean of Schedules to % SIO, LRT

Job ratio 10:1

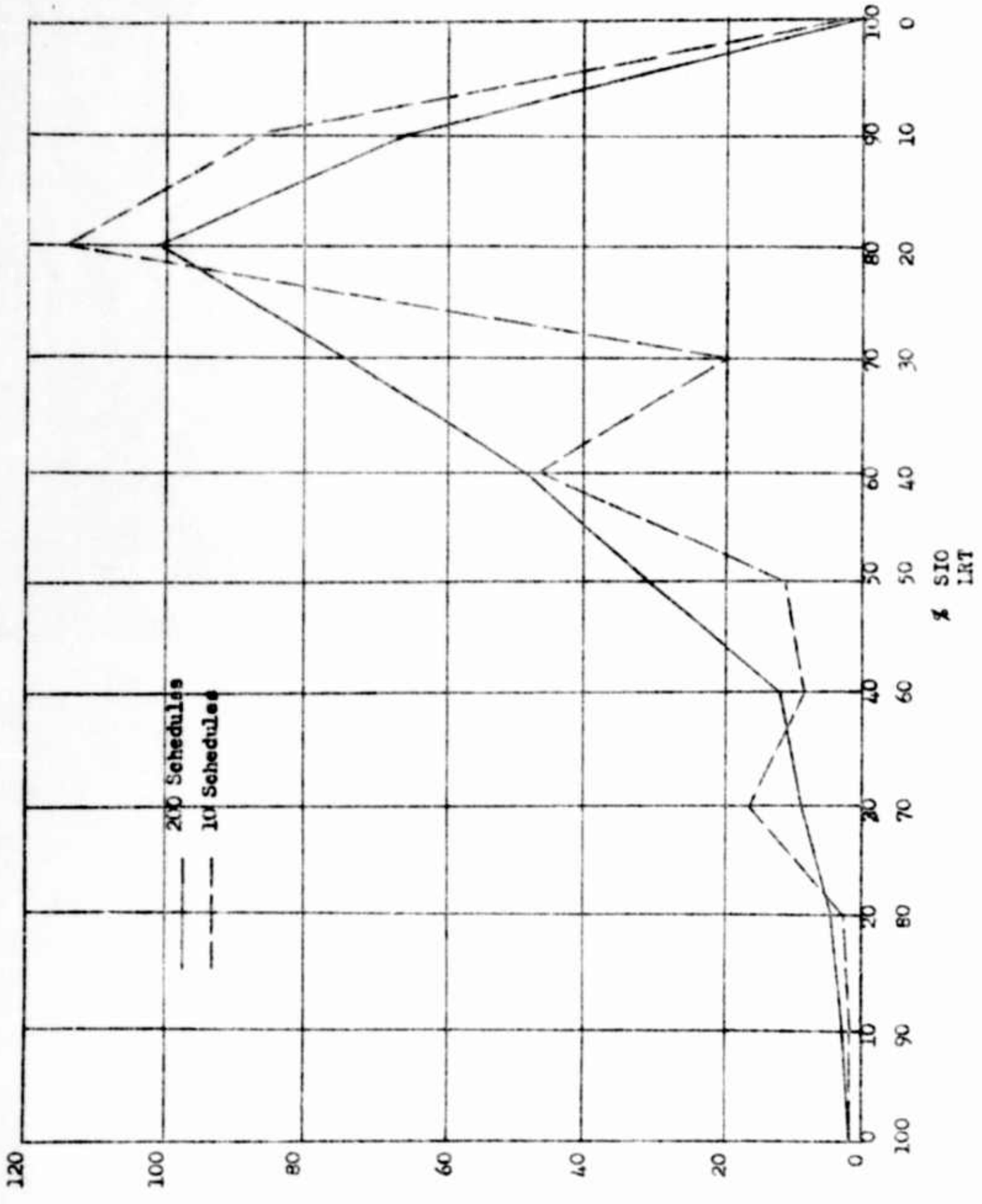


Figure 12

Relation of Mean of Schedules to % SIO, LRT

Job ratio 9:2

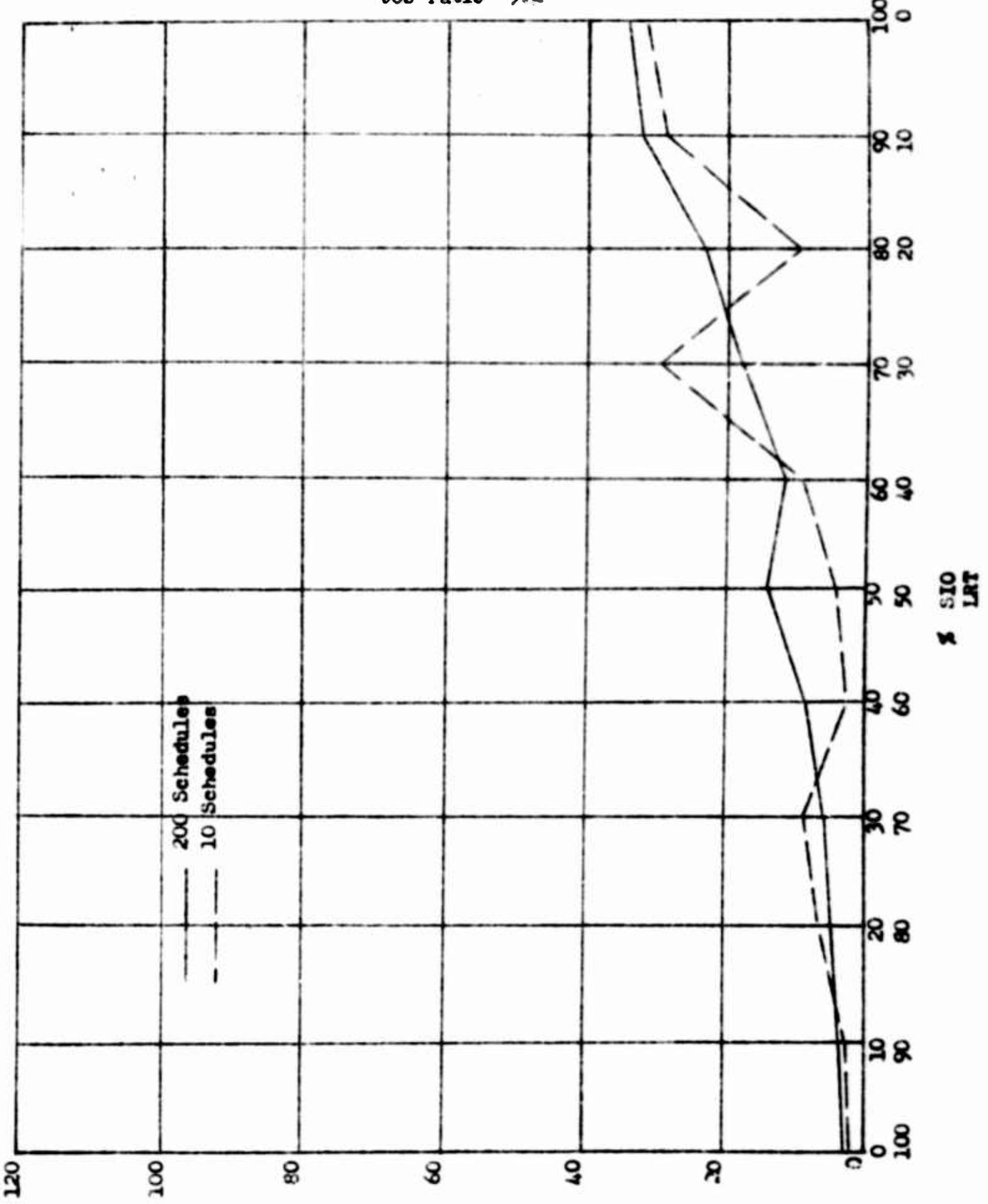


Figure 13

Relation of Mean of Schedules to % SIO, LRT
Job Ratio 8:5

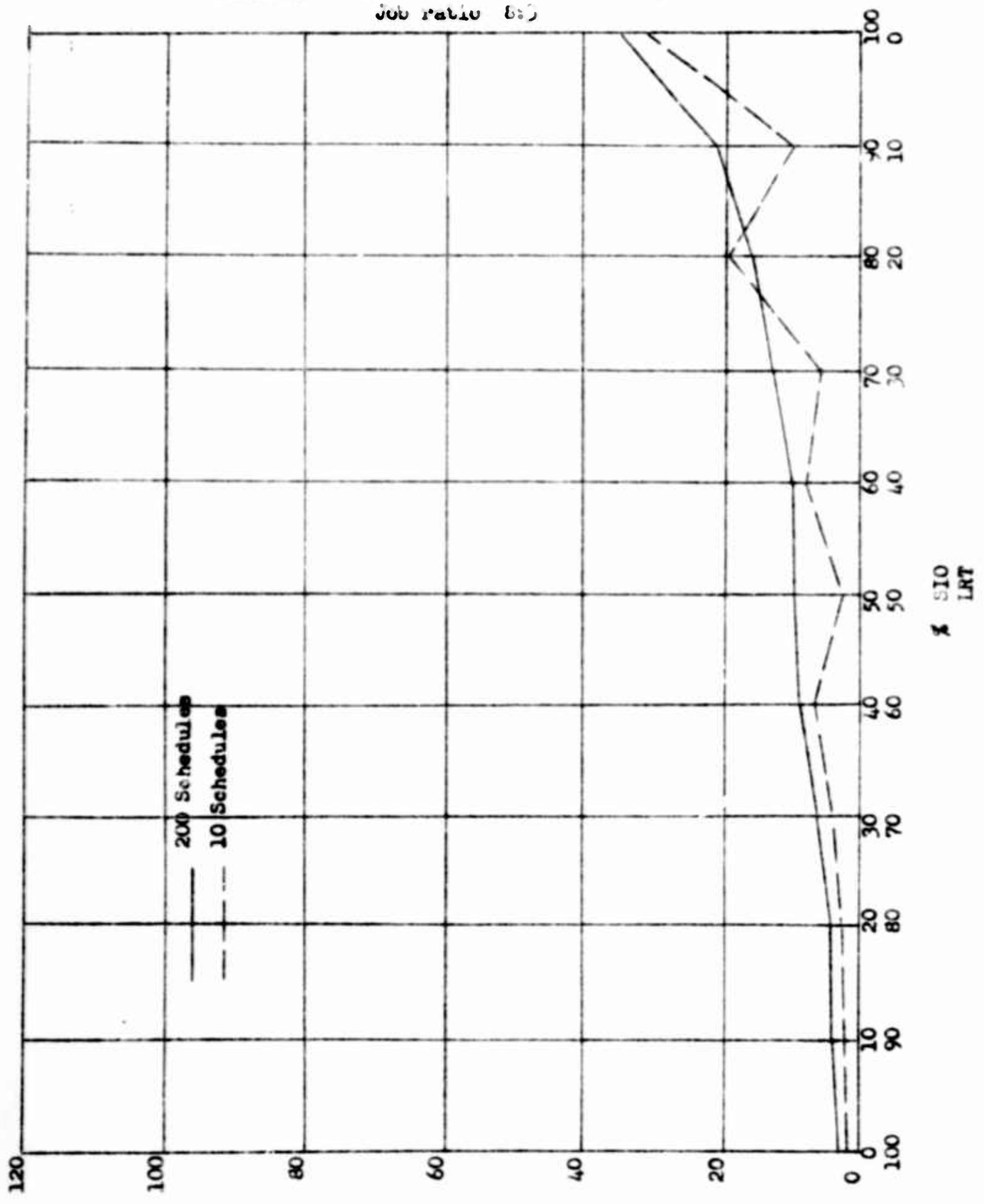


Figure 14

Relation of Mean of Schedules to % SIO, LRT
Job ratio 7:4

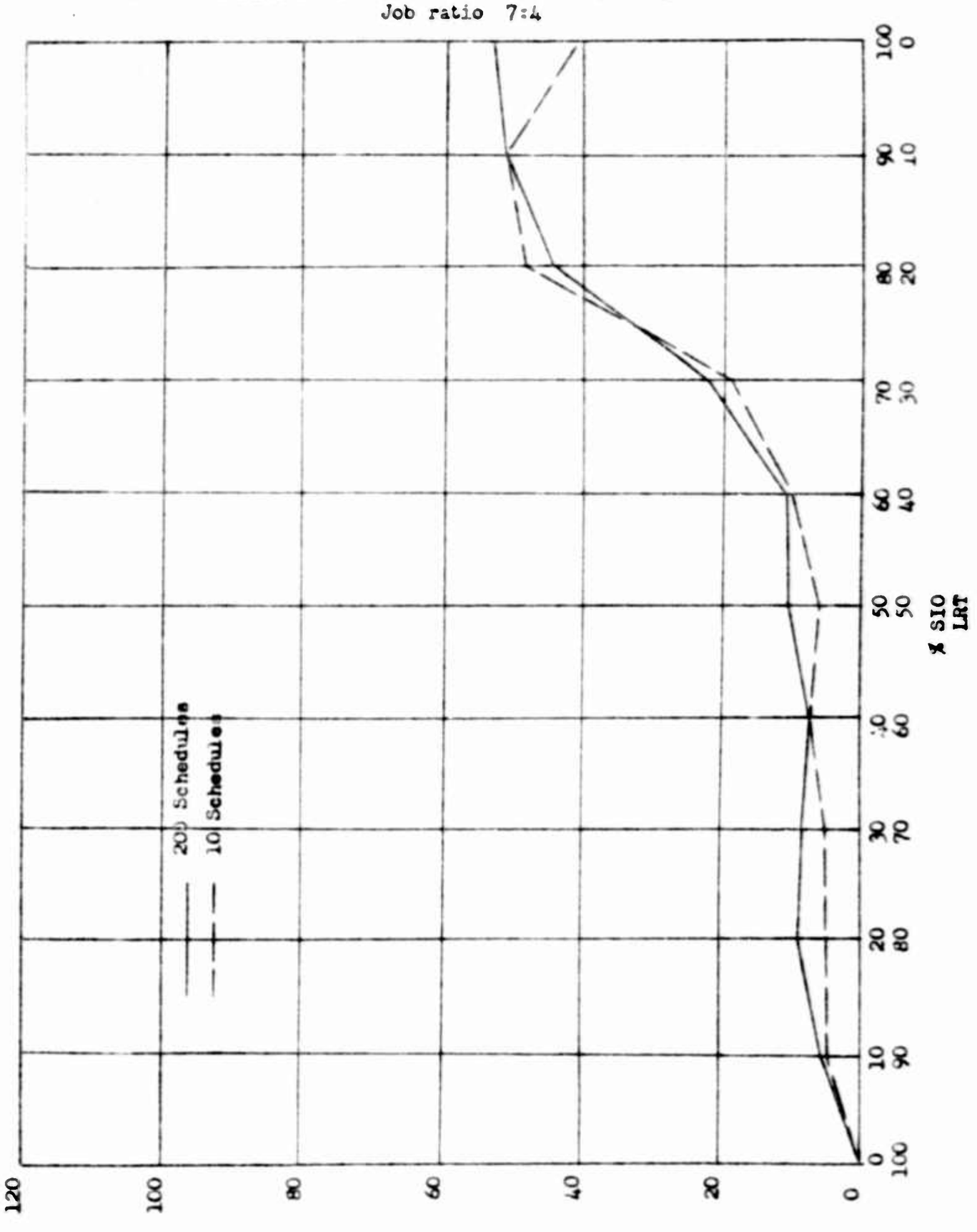


Figure 15

Occurrences of Optimum Schedule vs. % SIO, LRT
Job ratio 10:1

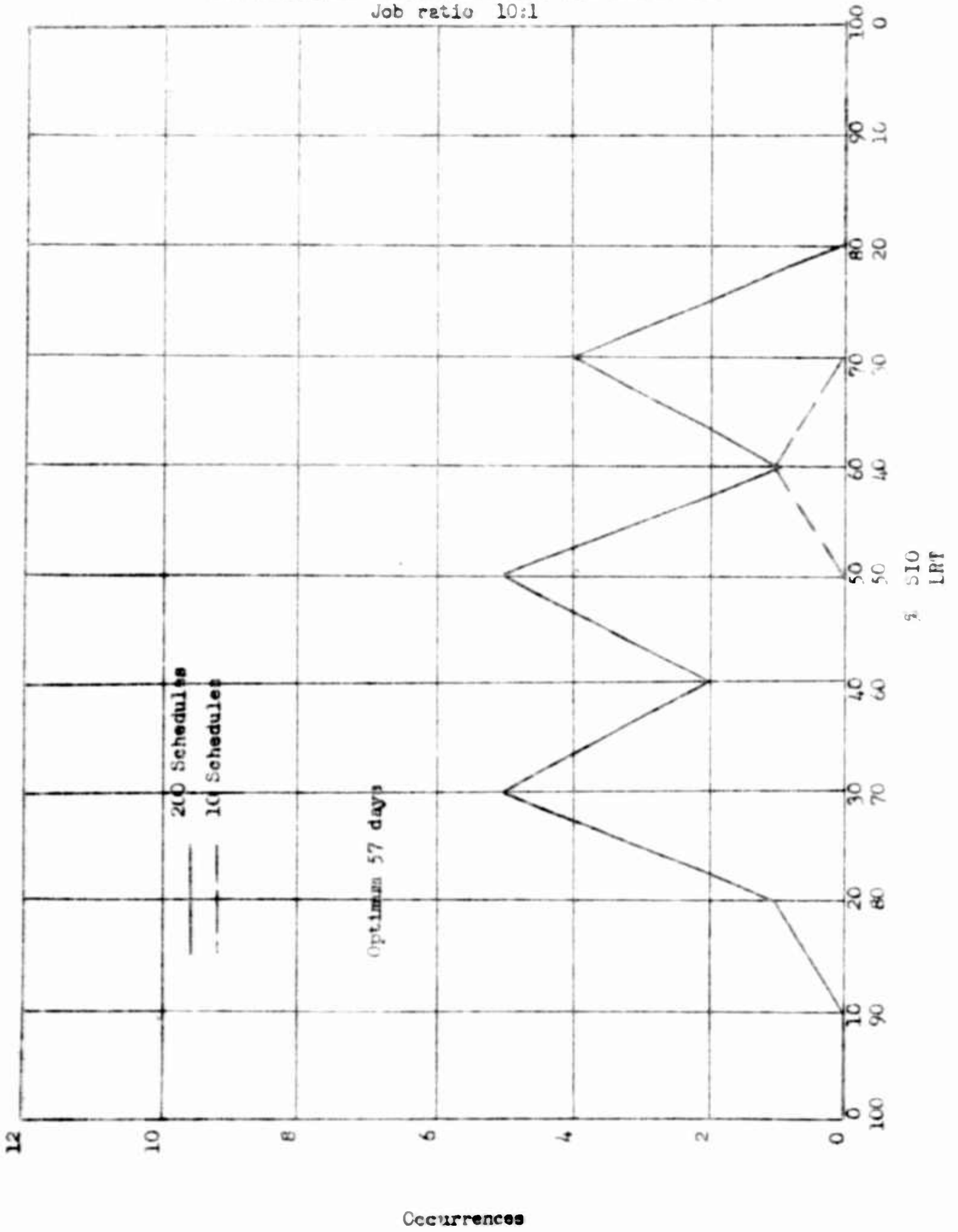


Figure 16

Occurrences of Optimum Schedule vs. % SIO, LRT

Job ratio 9:2

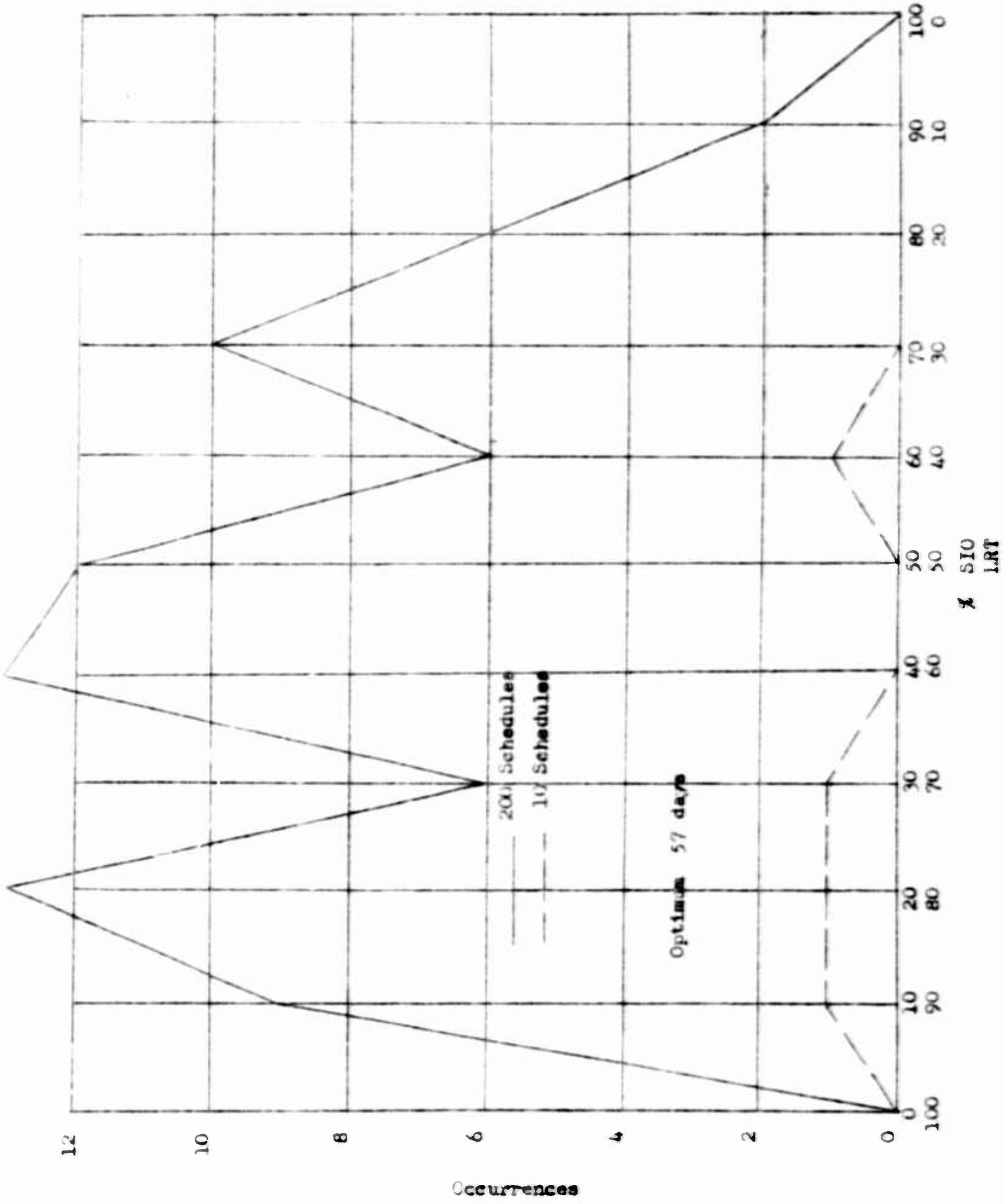


Figure 17

Occurrences of Optimum Schedule vs. % SIO, LRT

Job ratio 2:3

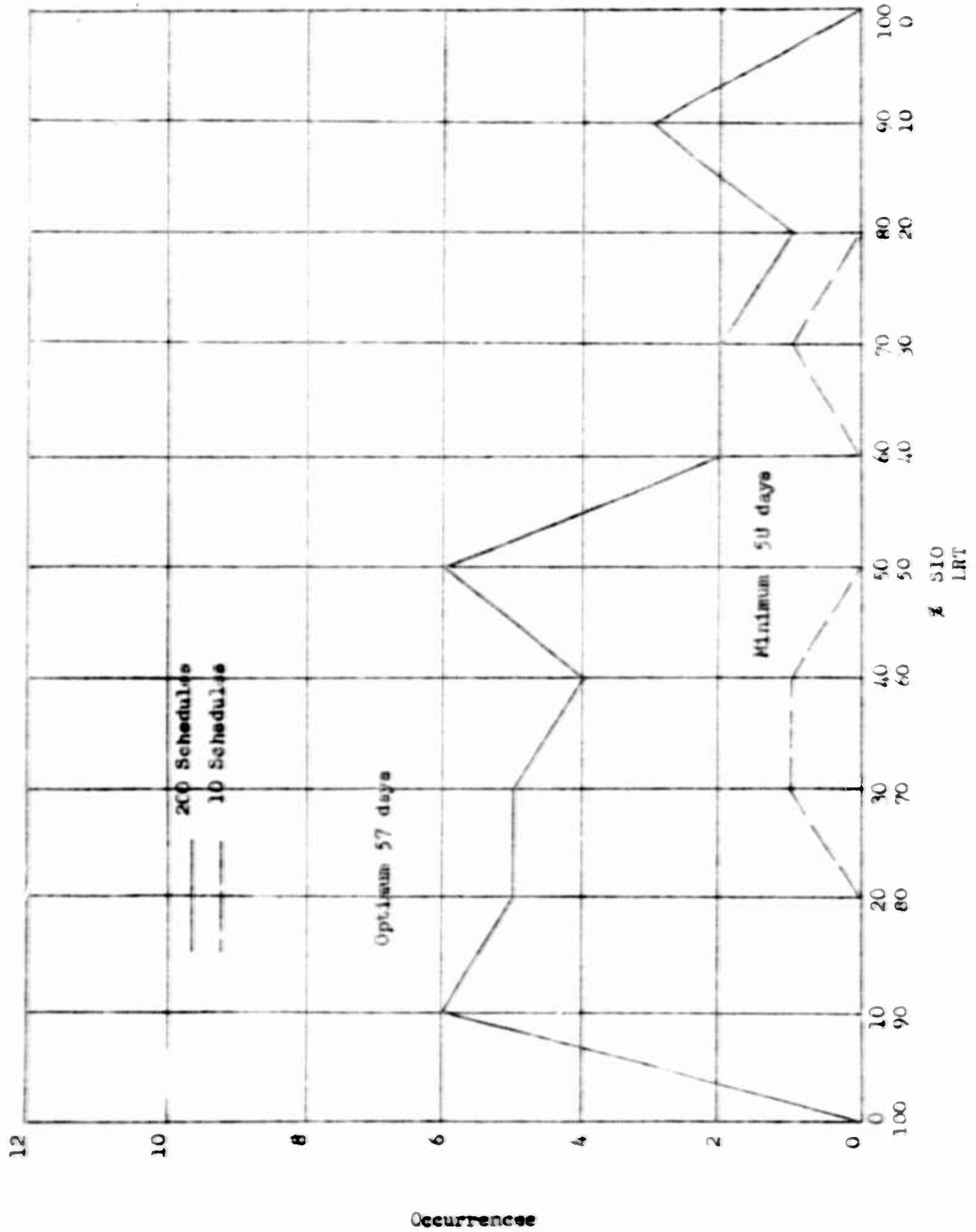
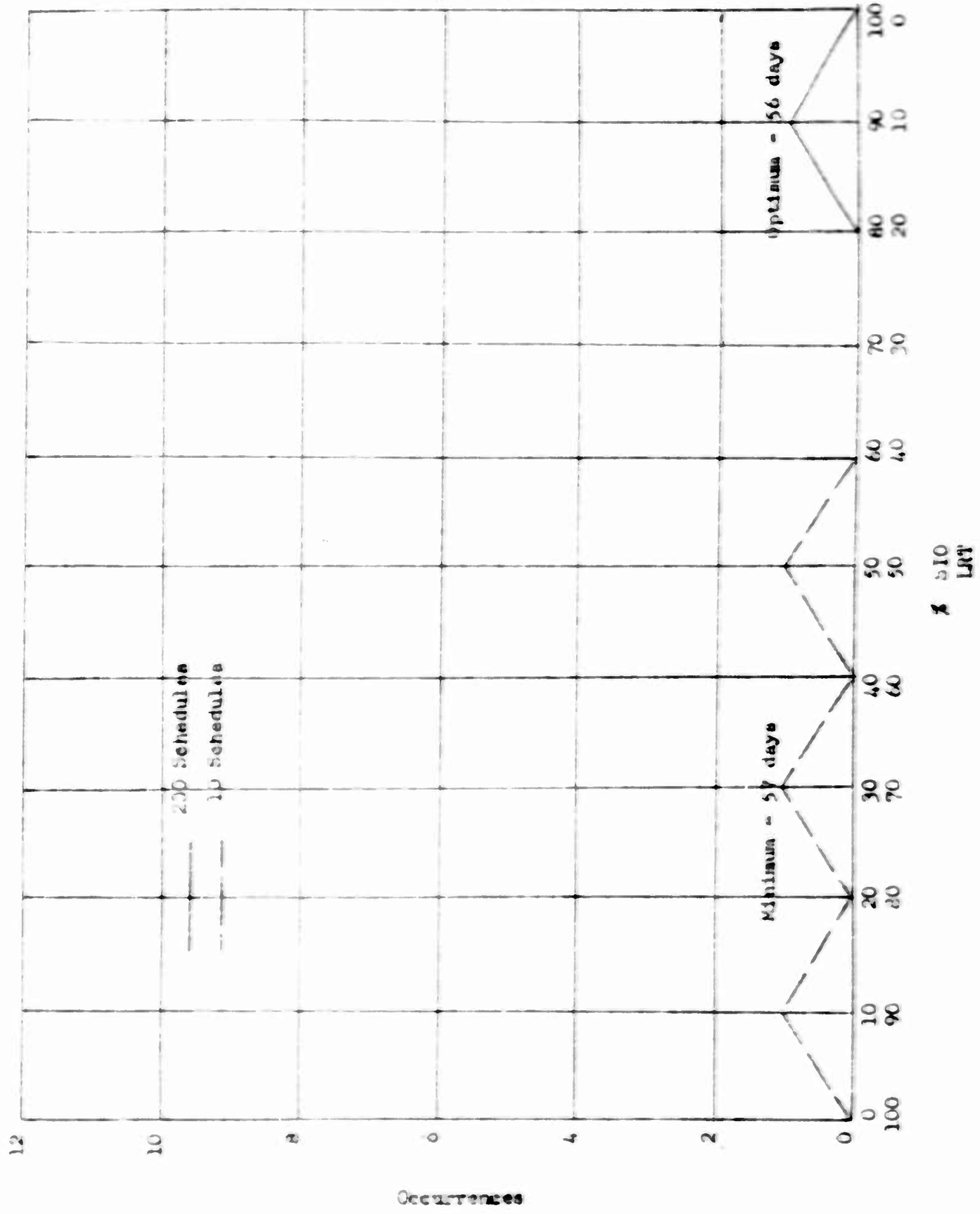


Figure 18

Occurrences of Optimum Schedule vs. % SIO, LRT

Job ratio 7:4



Various Combinations of LRT, SIO Within the Schedule For Problem 1 (10:1)

Sequence 1 - First half SIO 10 Schedules

Mean	67
Variance	0
Number of Optimum Schedules	0

Sequence 2 - First half LRT 10 Schedules

Mean	67
Variance	.6
Number of Optimum Schedules	0

Sequence 3 - Decreasing SIO Increasing LRT 100 Schedules

Mean	65.8
Variance	36.5
Number of Optimum Schedules	0

Sequence 4 - Increasing SIO Decreasing LRT 100 Schedules

Mean	61.4
Variance	20.0
Number of Optimum Schedules	6

Figure 19

3. DISCUSSION OF RESULTS

The results show that the %SIO, LRT used has a strong effect on the mean and variance of a series of schedules. In general, the mean of the series rises slightly between 0 and 50% SIO then begins to increase sharply as the per cent SIO increases. The variance increases as the per cent SIO increases but in several cases drops at 100% SIO. The data gives no clear proof that a certain ratio of LRT-SIO is in general superior or even superior in specific problems. For example, in problems 1, 2 and 3 the distribution of best schedules is relatively constant in the range 20 - 28 per cent SIO. Peaks do exist in the distributions, but they could occur due to chance. However, in problem 4 the best schedule does occur uniquely at 90% SIO.

To determine the effect of machining time ratio (ie., 10:1, 9:2, 8:3, 7:4) on ease of scheduling, the mean, the variance and the number of best schedules for the four problems will be compared. The following chart shows, for series of 200 schedules, the range of the means and of the variances, and shows the number of best schedules generated.

Problem Range	10:1	9:2	8:3	7:4
Range of Means of series	77-59=18	63-54=9	70-63=7	76-60=16
Range of Variance of series	100-0=100	35	33	49-0=49
Number of Best Schedules	16	77	34	1

The data does not show a clear relationship between the machining time ratio and the ease of scheduling. It would suggest that other properties of the numbers in the problem have an overriding effect.

The data in Figures 7-19 show clearly that the mean and variance of a series of ten schedules is very close to the values from a series of 200 schedules. The differences that do occur seem evenly distributed between + and - values. In problems 1, 2 and 3 the distributions of minimum schedules times in the series of ten schedules have their medians near the median of the series of 200. This situation did not occur in problem 4.

The results of varying the percentage SIO, LPT within a schedule are shown in Figure 19. A series of schedules generated with the first half of each schedule, scheduled by SIO and the second half by LPT is almost identical to a series in which the first half of each schedule is scheduled by LPT and the second half by SIO. Similarly there was little difference between schedules with SIO increasing from 10% to 90% and schedules with SIO decreasing from 90% to 10%. The difference did favor increasing SIO.

4. SIMPLE LEARNING

The results of the previous tests suggested a simple form of learning might be effective. In the four problems examined, the mean and variance of a series of ten schedules were very close to the mean and variance of the series of two hundred schedules at all ratios of SIO to LRT. Also in problems 1, 2 and 3 the distribution of minimum schedule times in the short series had its median at approximately the same location as the median of the distribution for the long series. Thus we can gain much information about a specific problem from a series of short sequences of schedules at various combinations of SIO and LRT.

The learning uses this fact to determine an optimum ratio of LRT to SIO as follows:

1. Generate nine series of ten schedules each with combinations of SIO and LRT, starting at 10:90 and changing in eight steps of 10 to 90:10.
2. The best ratio is determined by finding the series with the lowest schedule time. In case of a tie the one with the most "best schedules" is chosen. If there is still a tie the series with the lowest variance is chosen as the first optimum ratio.
3. The program then concentrates on an area 20% above and 10% below the optimum per cent LRT. Over this reduced range nine series of 20 schedules are generated.
4. As in step two an optimum ratio is determined. This final optimum ratio is used to generate an additional 100 schedules.

The result of its application to the four problems of the previous section are shown in Figures 20-23.

Simple Learning

Problem 1: Machining Time Ratio 10:1

Step 1
10 Schedules

%SIO	10	20	30	40	50	60	70	80	90
Mean	59	58	58	62	66	61	70	73	78
Variance	1	1	1	33	136	27	103	139	21
Best Schedule	58	57	55	55	55	56	59	57	76
Occurences	4	2	1	1	1	2	2	1	1

Best ratio 30% SIO

Step 2
20 Schedules

%SIO	13.3	16.7	20	23.3	26.7	30	33.3	36.7	40
Mean	59	59	58	59	59	58	60	59	59
Variance	5	4	2	4	11	2	17	6	13
Best Schedule	56	56	56	56	56	55	56	57	56
Occurences	2	2	1	2	1	1	3	5	1

Best ratio 30% SIO

Step 3
100 Schedules

%SIO	30
Mean	60
Variance	32
Best Schedule	55
Occurences	2

Figure 20

Simple Learning

Problem 2: Machining Time Ratio 9:2

Step 1
10 Schedules

%SIO	10	20	30	40	50	60	70	80	90
Mean	54	53	54	55	54	57	58	57	59
Variance	4	[5]	1	2	3	6	40	15	18
Best Schedule	51	[50]	53	53	52	55	52	[50]	55
Occurence	1	[1]	2	2	1	2	1	[1]	1

Best ratio 20% SIO

Step 2
20 Schedules

%SIO	3.3	6.7	10	13.3	16.7	20	23.3	26.7	30
Mean	54	54	54	54	54	53	54	55	54
Variance	2	1	2	2	2	9	2	6	3
Best Schedule	53	53	[50]	53	53	[50]	53	51	51
Occurence	11	9	1	7	7	[3]	6	1	1

Best ratio 20% SIO

Step 3
100 Schedules

%SIO	20
Mean	55
Variance	5
Best Schedule	50
Occurence	3

Figure 21

Simple Learning

Problem 3: Machining Time Ratio 8:3

Step 1
10 Schedules

%SIO	10	20	30	40	50	60	70	80	90
Mean	63	62	64	61	62	61	62	62	67
Variance	1	7	2	8	11	8	8	4	25
Best Schedule	62	58	62	57	59	57	60	59	61
Occurence	4	1	1	2	4	1	2	1	1

Best ratio 40% SIO

Step 2
20 Schedules

%SIO	23.3	26.7	30	33.3	36.7	40	43.3	46.7	50
Mean	63	62	62	63	62	61	62	62	62
Variance	4	4	6	3	7	6	5	7	6
Best Schedule	58	57	57	59	58	57	58	58	58
Occurence	1	1	1	1	1	3	1	2	2

Best ratio 40% SIO

Step 3
100 Schedules

%SIO	40
Mean	62
Variance	8
Best Schedule	57
Occurence	1

Figure 22

Simple Learning

Problem 4: Machining Time Ratio 7:4

Step 1
10 Schedules

%SIO	10	20	30	40	50	60	70	80	90
Mean	61	62	61	63	62	63	62	64	68
Variance	4	10	2	16	5	27	11	28	62
Best Schedule	60	59	59	59	60	57	59	59	59
Occurence	7	1	1	1	2	2	2	2	1

Best ratio 60% SIO

Step 2
20 Schedules

%SIO	43.3	46.7	50	53.3	56.7	60	63.3	66.7	70
Mean	61	63	62	62	63	62	63	62	64
Variance	9	10	12	16	42	9	16	23	20
Best S Schedule	57	59	57	57	56	57	58	55	58
Occurence	2	1	1	1	1	1	1	1	1

Best ratio 66.7% SIO

Step 3
100 Schedules

%SIO	66.7
Mean	64
Variance	22
Best Schedule	56
Occurence	1

Figure 23

5. DISCUSSION OF RESULTS FOR SIMPLE LEARNING

To judge the effectiveness of the simple learning routine the results will be compared with the results of 50:50 SIO, LRT series.

It may be seen from an examination of problems 1 and 4 that the learning routine may give an optimum ratio SIO:LRT either below or above 50:50 (30:70 and 66.7:33.3 respectively). Since the series mean and variance tend to increase with the per cent SIO the learning routine may give a series mean and variance either higher or lower than that given by the 50:50 ratio.

As part of the introductory work with the program, 400 schedules were generated for problem 1 with a 50:50 ratio. In this series the optimum time of 55 occurred 6 times. In the 370 schedules in the learning routine for problem 1, exactly 6 optimum schedules were also generated. Therefore, the use of learning in the program does not necessarily give a lower mean, a lower variance, or a higher number optimum schedules.

It is interesting to note that the program with learning did find a new minimum for problem 4. It is not possible to accept or reject this learning on the basis of the present data. There is a possibility that it would be more efficient than the 50:50 ratio on a larger problem or more efficient on the 6X6 if the number of schedules generated at each stage were reduced.

Appendix A

Gantt Charts of Optimum Schedules

PROBLEM 1 - 10:1

Permanent Schedule

Day	Machine 1	2	3	4	5	6
1	0	2	3	0	0	0
2	0	2	3	0	0	0
3	0	2	3	0	0	0
4	0	2	3	0	0	0
5	0	2	3	0	0	0
6	0	2	1	3	0	0
7	1	2	0	3	0	0
8	1	2	0	3	0	0
9	1	4	2	3	0	0
10	0	4	2	0	0	3
11	0	4	2	0	0	3
12	0	4	2	0	0	3
13	0	4	2	0	0	3
14	4	6	5	0	2	3
15	4	6	5	0	2	3
16	4	6	5	0	2	3
17	4	1	5	6	2	3
18	4	1	5	6	2	0
19	3	1	5	6	2	0
20	3	1	5	0	2	6
21	3	1	5	0	2	6
22	3	1	5	0	2	6
23	3	5	4	1	2	6
24	3	5	4	1	0	6
25	3	5	4	1	0	6
26	3	0	4	1	5	6
27	3	0	4	1	5	6
28	0	3	0	1	5	6
29	6	0	0	1	5	2
30	6	0	0	4	5	2
31	6	0	0	4	3	2
32	6	0	0	4	3	2
33	6	0	0	0	3	2
34	6	0	0	0	3	2
35	6	0	0	0	3	2
36	6	0	0	0	3	2
37	6	0	0	0	3	2
38	6	0	0	0	4	2
39	2	0	0	0	4	1
40	2	0	0	0	4	1
41	2	0	0	0	4	1
42	2	0	0	0	4	5
43	2	0	0	0	4	5
44	2	0	0	0	4	5
45	2	0	0	0	4	5
46	2	0	0	0	6	4
47	2	0	0	0	6	4
48	2	0	0	0	6	4
49	5	0	0	2	6	4
50	5	0	6	2	1	4
51	5	0	0	2	1	4
52	0	0	0	2	1	4
53	0	0	0	5	1	4
54	0	0	0	0	1	4
55	0	0	0	0	1	0

Permanent Schedule

PROBLEM 2 - 9:2

Day	Machine 1	2	3	4	5	6
1	0	4	1	0	0	0
2	0	4	1	0	0	0
3	4	6	1	0	0	0
4	4	6	1	0	0	0
5	4	6	1	0	0	0
6	0	6	1	0	0	0
7	1	6	3	0	0	0
8	1	6	3	0	0	0
9	1	6	3	0	0	0
10	1	2	3	6	0	0
11	1	2	3	6	0	0
12	1	2	3	0	0	6
13	1	2	4	3	0	6
14	1	2	4	3	0	0
15	1	2	4	3	0	0
16	6	2	4	3	0	0
17	6	2	4	3	0	0
18	6	2	4	3	0	0
19	6	1	4	0	0	3
20	6	1	2	4	0	3
21	6	1	2	4	0	3
22	6	1	5	4	2	3
23	6	1	5	4	2	3
24	6	1	5	4	2	0
25	3	1	5	4	2	0
26	3	1	5	4	2	0
27	3	1	5	4	2	0
28	3	5	0	1	6	2
29	3	5	0	1	6	2
30	3	5	0	1	6	0
31	3	0	6	1	5	0
32	3	0	6	1	5	0
33	2	3	6	1	5	0
34	2	3	6	0	4	5
35	2	3	6	0	4	5
36	2	0	6	0	4	5
37	2	0	6	0	4	5
38	5	0	6	2	3	1
39	5	0	6	2	3	1
40	5	0	0	2	3	1
41	5	0	0	2	3	1
42	0	0	0	2	3	1
43	0	0	0	2	3	4
44	0	0	0	2	3	4
45	0	0	0	5	3	4
46	0	0	0	5	1	4
47	0	0	0	5	1	4
48	0	0	0	5	1	4
49	0	0	0	5	0	4
50	0	0	0	0	0	4

Permanent Schedule — PROBLEM 3 - 6:3

Day	Machine 1	2	3	4	5	6
1	0	6	1	0	0	0
2	0	6	1	0	0	0
3	0	6	1	0	0	0
4	0	2	1	6	0	0
5	0	2	1	6	0	0
6	0	2	1	6	0	0
7	0	2	1	6	0	0
8	0	2	1	6	0	0
9	1	2	3	6	0	0
10	1	2	3	6	0	0
11	1	2	3	6	0	0
12	1	0	3	0	0	6
13	1	4	3	0	0	6
14	0	4	3	0	0	6
15	0	4	2	3	0	6
16	0	4	2	3	0	6
17	6	4	2	3	0	0
18	6	4	2	3	0	0
19	6	4	2	3	0	0
20	6	4	2	3	0	0
21	4	1	2	3	6	0
22	4	1	5	3	6	0
23	4	1	5	0	6	3
24	4	1	5	0	6	3
25	4	0	5	1	6	3
26	4	0	5	1	2	3
27	4	0	5	1	2	3
28	4	0	5	1	2	3
29	3	0	5	1	2	0
30	3	5	4	0	2	0
31	3	5	4	0	2	0
32	3	5	4	0	0	2
33	3	5	6	4	0	2
34	3	0	6	4	5	2
35	3	0	6	4	5	1
36	3	0	0	4	5	1
37	2	3	0	4	0	1
38	2	3	0	4	0	1
39	2	3	0	0	4	1
40	2	0	0	0	4	1
41	2	0	0	0	4	1
42	2	0	0	0	4	5
43	0	0	0	2	4	5
44	0	0	0	2	1	5
45	0	0	0	2	1	5
46	0	0	0	2	1	5
47	0	0	0	2	1	5
48	5	0	0	2	1	4
49	5	0	0	2	1	4
50	5	0	0	0	1	4
51	0	0	0	5	1	4
52	0	0	0	5	3	4
53	0	0	0	5	3	0
54	0	0	0	0	3	0
55	0	0	0	0	3	0
56	0	0	0	0	3	0
57	0	0	0	0	3	0

Permanent Schedule

PROBLEM 4 - 7:4

Day	Machine 1	2	3	4	5	6
1	0	6	5	0	0	0
2	0	6	5	0	0	0
3	0	6	5	0	0	0
4	0	6	5	0	0	0
5	0	6	5	0	0	0
6	0	6	5	0	0	0
7	0	4	1	6	0	0
8	0	4	1	6	0	0
9	0	4	1	6	0	0
10	0	4	1	6	0	0
11	0	4	1	0	0	6
12	0	4	1	0	0	6
13	4	5	3	0	0	6
14	4	5	3	0	0	6
15	4	5	3	0	0	6
16	4	5	3	0	0	6
17	4	5	3	0	0	6
18	4	5	3	0	0	0
19	6	5	4	3	0	0
20	6	2	4	3	5	0
21	6	2	4	3	5	0
22	6	2	4	3	5	0
23	6	2	4	3	5	0
24	6	2	4	3	5	0
25	6	2	4	0	5	3
26	0	2	0	4	5	3
27	1	0	2	4	0	3
28	1	0	2	4	0	3
29	1	0	2	4	6	3
30	1	0	2	4	6	5
31	1	0	2	0	6	5
32	1	0	2	0	6	5
33	1	0	2	0	6	5
34	3	1	0	0	6	5
35	3	1	6	0	2	0
36	3	1	6	0	2	0
37	3	1	6	0	2	0
38	3	0	6	1	2	0
39	5	3	6	1	4	2
40	5	3	6	1	4	2
41	5	3	0	1	4	2
42	5	3	0	1	4	2
43	5	3	0	1	4	2
44	5	3	0	1	4	2
45	2	0	0	5	3	1
46	2	0	0	5	3	1
47	2	0	0	5	3	1
48	2	0	0	5	3	1
49	2	0	0	5	3	1
50	2	0	0	5	3	4
51	2	0	0	5	3	4
52	0	0	0	2	1	4
53	0	0	0	2	1	4
54	0	0	0	2	1	0
55	0	0	0	2	1	0

CHAPTER III

PROBABILISTIC LEARNING COMBINATIONS OF LOCAL JOB SHOP SCHEDULING RULES

Learning Process Used

1. INTRODUCTION

As an extension of the basic job shop scheduling program (discussed in Chapter I) a learning segment was developed and studied as a means of improving the efficiency of the program for producing "good" schedules. Three learning processes were designed and tested but only one of these gave sufficient promise of success to warrant extended investigation.

The learning process is readily adaptable to any number of rules. Though six rules were incorporated in the scheduling program, only three rules were used in the investigation of the learning process. These were the shortest imminent operation, longest remaining time, and machine slack rules. Computer runs using various combinations of more than three rules indicated that a great number of schedules would be required to generate good schedules because of the greater number of possible sequences of rule choices. The three rules used seem to provide for a complementary set of rules which should intuitively be necessary for arriving at a "good" sequence of decisions. Machine slack has as its primary objective the avoidance of idle time on critical machines. The longest remaining time rule, in some sense, attempts to expedite those goods which have the least slack available. The job slack rule might be used as a substitute for this rule. The only difference between these rules is that job slack considers the job's slack relative to the number of operations remaining to be performed. The shortest imminent operation rule primarily expedites short operation goods which moves as many goods as possible in as short a period of time as possible.

2. DESCRIPTION OF THE LEARNING PROCESS

The learning process is characterized by an unbiased starting position and the designation of time periods during which decisions are made on each machine. The length of each time period and the number of time periods are prescribed for the program. The last time period is understood to include all ending decisions even though the period length may be greater than that used in all preceding periods. That is, if time periods are prescribed as being 200 units of time in length and the last time period commences at the 800th time unit, all decisions made after the 1000th time unit would still be regarded as being made in the last time period. Time period lengths do not necessarily need to be uniform. However, in all investigations, they were prescribed as being equal.

The program initially generates some prescribed number of schedules with an unbiased starting position which implies that the probability of selecting any of the three decision rules is equal at all decision points. For each schedule, a record is kept of the rule selected at each decision point and the time period in which the decision point is located. Each time a superior schedule is generated, the sequence of decisions which produces the schedule is stored as the most recent best standard.

After generating the prescribed number of schedules, control is passed to the learning segment. Each decision point in the decision probability matrix which occurs in the first time period is then biased to certainty to select the rule prescribed for the decision point in the most recent best standard. All decision points in later time periods are left unbiased.

The scheduling program then generates another prescribed number of schedules. The sequences of goods scheduled during the first time period

will be identical for each of these schedules and only decisions following time period one vary. Again, each time a superior schedule is generated, the sequence of decisions is stored as the most recent best standard.

The rules prescribed in the standard which occur in the first two time periods are then set to certainty in the decision probability matrix and decision points in later time periods are maintained in an unbiased position. This process continues until the last time period is the only one with varying decisions possible. At each step through the time periods, the number of varying decision points is decreased.

After all time periods have been considered, the second step in the learning process takes place. Each decision point in the decision probability matrix is then biased by a fixed amount in the direction of choosing the rule prescribed for the point in the standard sequence. That is, if the standard sequence specified that the SIO rule was to be selected at the 5th decision point on a particular machine, a 50% probability might be placed on selecting this rule and the other two rules would have probabilities of 25% each. The process of generating schedules through time periods (as previously described) would then be repeated. Convergence toward a unique decision vector can be accommodated by strengthening the bias which is placed on the standard sequence decisions at the end of each progression through all time periods.

3. VERBAL STEP DESCRIPTION OF LEARNING PROGRAM

1. Generate a schedule ($i = 1,\ldots,n$)

   a. Each time a decision is made, store the time period and rule choice for the decision point.

   b. If the schedule is better than the previous standard, go to c. If not, $i \leftarrow i + 1$ if $i \neq n$ and go to 1. If $i = n$, go to 2.

2. Bias to certainty the sequence of rules through time period t.

   a. If t is the last time period, go to 3.

   b. Go to 1, $t \leftarrow t + 1$.

3. Bias each decision point in the decision probability matrix toward the rule specified for the point in the standard sequence by some predetermined amount. Go to 1.

Discussion of results: In all tests of the learning process, it was found that when the bias for the standard sequence became too high, the generating of better sequences became severely restricted. With a bias of 50% or 60% toward the best rule at each decision point, much more effective learning was observed.

Also, since fewer decision points are varying when later time periods are under consideration (i.e., decisions made in early time periods are set to certainty) a fewer number of schedules needs to be generated to find a better sequence than in early time periods when many (or all) decision points are varying. It was observed that the generating of better schedules with all decisions varying seldom occurred. A few of the combinations of number of time periods, and loops/time period are shown below for the 20X5X5 and 10X10X10 problems along with their results.

20X5X5 Problem

No. of Time Periods	Schedules/ Time Period	Total	Total Loops	Best Schedule	Running Time
3	5	15	105	1326	42 min.
5	3	15	105	1257	40
5	5	25	100	1227	37
5	(8,6,3,2,2)	21	85	1221	33
7	(1,5,5,5,2,2,1)	21	105	1254	40
10	2	20	120	1236	45
5	3	15	105	1046	38
5	(5,4,3,2,1)	15	105	994	34
5	(1,1,5,5,3)	15	105	1080	26

The above results appear to indicate that the use of a variable number of schedules per time period with the number decreasing as later time periods are considered is the best approach to the problem.

4. TEST RESULTS

Learning with the 6X6X6 problem was very poor. This might be explained by the fact that a "good" schedule (only 2 or 3 time units above the known minimum of 55) was always generated while all decision points were unbiased. Continued search was unable to get below this "good" schedule. As the bias on the standard sequence of rules was strengthened, a smaller number of "bad" schedules was generated than by the purely random (all decision points equiprobable), non-learning process. From the number of schedules that were generated during the 6X6X6 test runs, considering both learning and non-learning runs, it should be expected that at least one schedule achieving the minimum of 55 would occur based on previous experience using only the SIO and LRT rules. The addition of the machine slack rule may therefore make it impossible (or extremely difficult) to achieve the minimum. Since results from tests using the 6X6X6 problem were not significant, graphical plots of the results have not been made.

Typical results achieved in test runs of the learning process using the 20X5X5 problem are plotted in Charts 1-4. Chart 5 shows the results of generating 100 schedules with all rules (MS, SIO, LRT) having equal probability of selection at all decision points (i.e., non-learning). Chart 6 shows the cumulative number of schedules found with total schedule times less than X hours both for the learning and non-learning processes. The results appear to indicate that progressive learning is taking place. Also, it was observed that learning progressed very rapidly during the early stages of computer runs and decreased during the later stages

Typical results achieved in test runs of the learning process using the 10X10X10 problem are plotted in Charts 7-8. Chart 9 shows the results for 100 schedules and non-learning. Chart 10 shows the cumulative number of schedules found with total schedule times less than X hours both for the learning and non-learning processes. Again, results appear to indicate that progressive learning is taking place.

At the bottom of each chart, the biases placed on the standard sequence of rules is indicated. The remaining two rules were made equiprobable. At the start of each of the new biasing periods, a greater variability is observed since all decision points are allowed to vary. As scheduling approaches the last time period (the end of each of the biasing periods), the variability is reduced since a unique sequence of decisions is being made in earlier time periods and only decisions in the late time periods are allowed to vary.

CHART 1

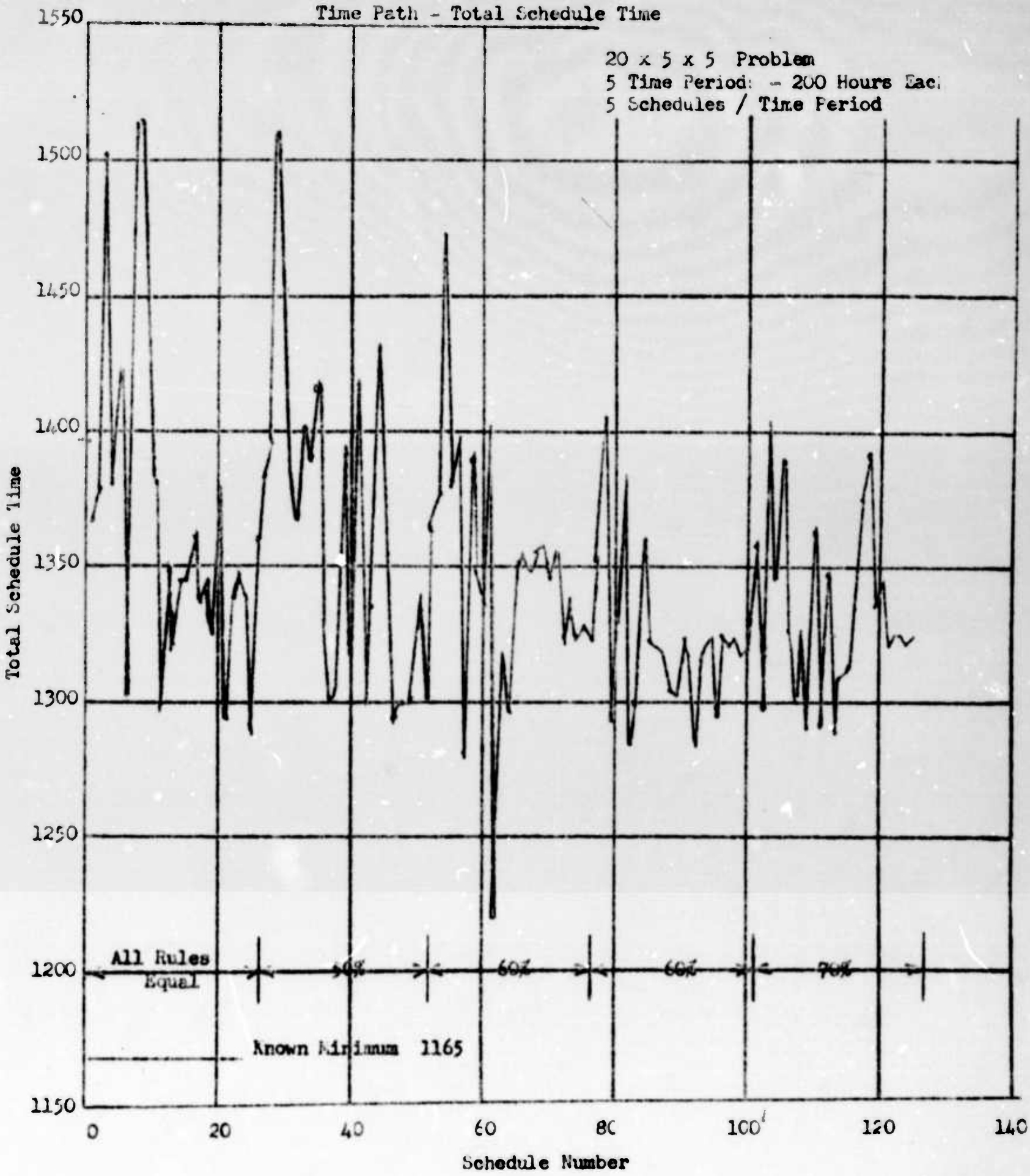

Time Path - Total Schedule Time
20 x 5 x 5 Problem
5 Time Periods - 200 Hours Each
5 Schedules / Time Period
1550
1500
1450
1400
1350
1300
1250
1200
1150
Total Schedule Time
All Rules Equal
60%
60%
70%
Known Minimum 1165
0
20
40
60
80
100
120
140
Schedule Number

CHART 2

Time Path - Total Schedule Time

20 x 5 x 5 Problem
10 Time Periods - 100 hours each
2 Schedules / Time Period
Biasing toward standard as shown

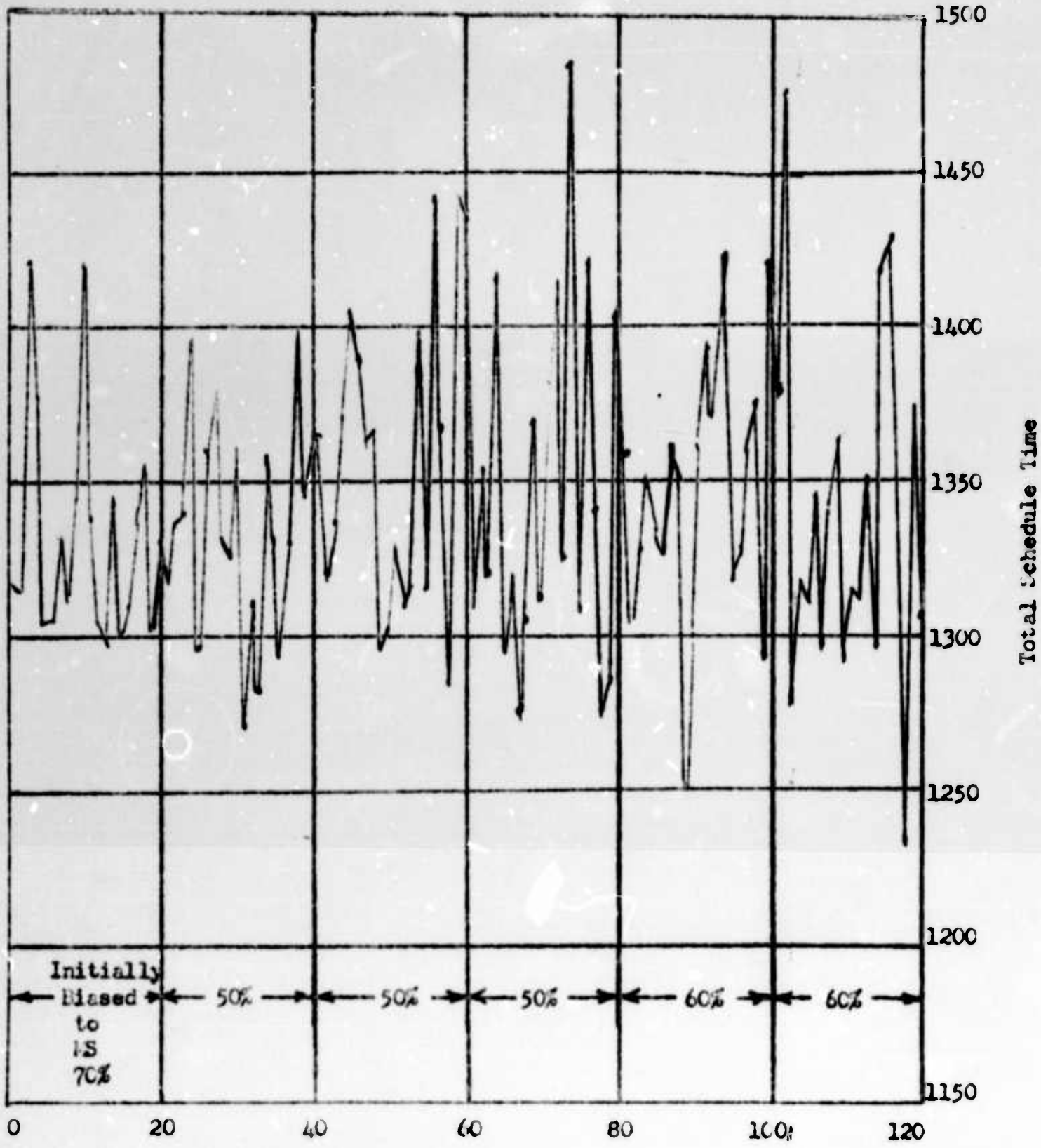


CHART 3

Time Path - Total Schedule Time

20 x 5 x 5 Problem
5 Time Periods - 200 hours each
8,6,3,2,2 Schedules Time Period 1-5 Resp.
Biasing toward Standard Rules as shown
* Initial Standard taken from final standard determined in Run on Chart 2

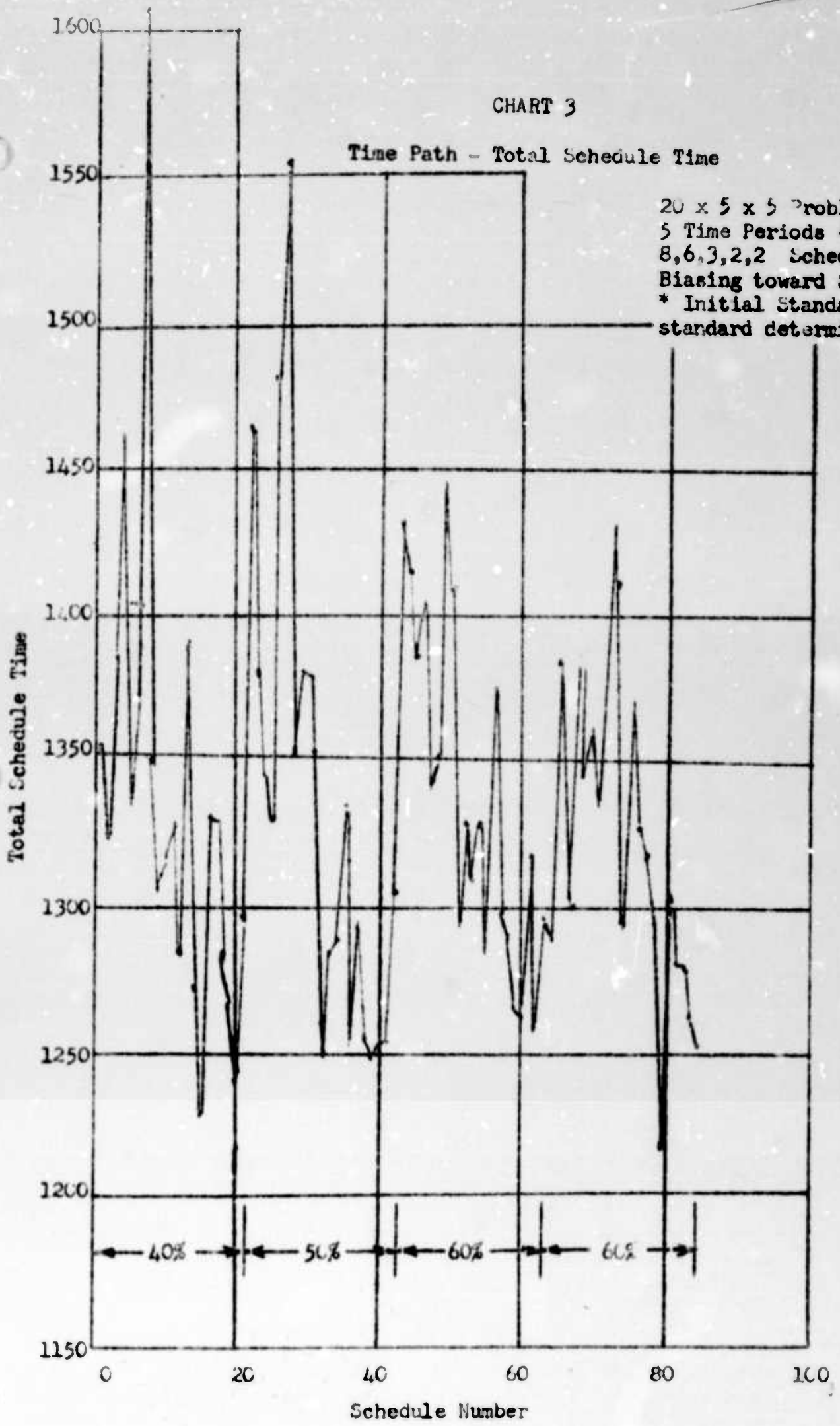


CHART 4

Time Path - Total Schedule Time

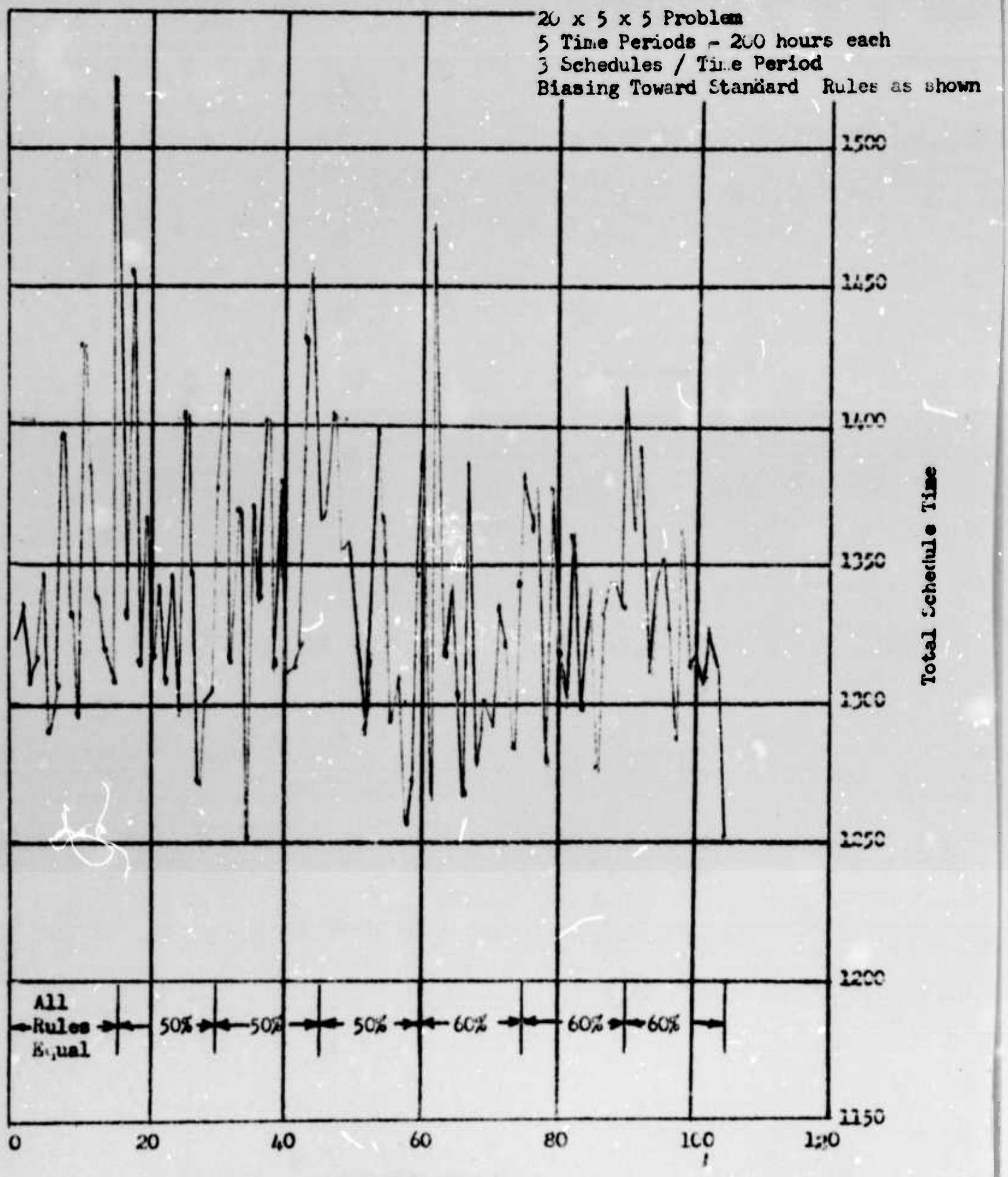


CHART 5

Time Path - Total Schedule Time

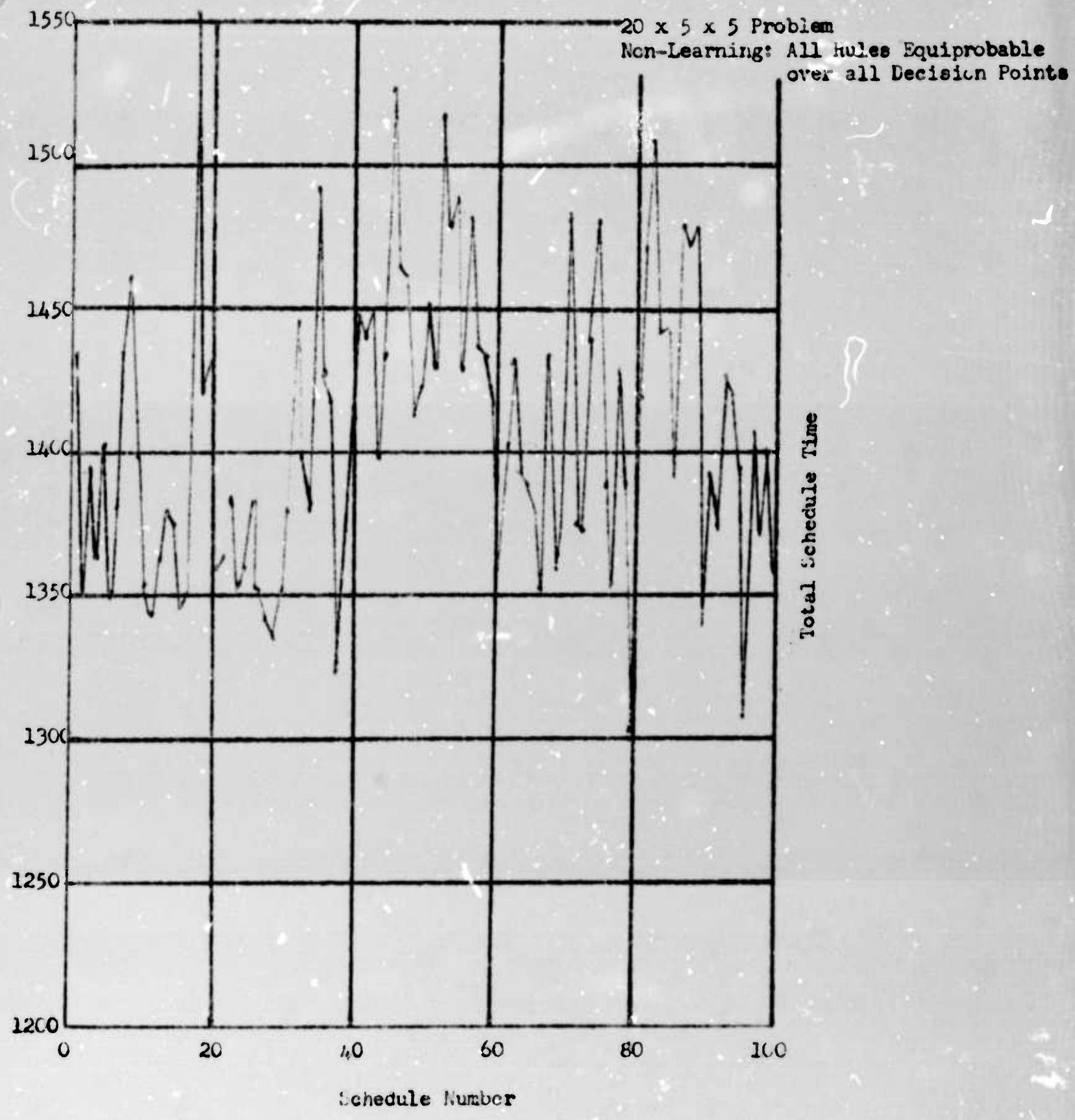


Chart 6

Cumulative Distribution Diagram
20×5×5 Problem
Learning Curve - Constructed from Chart 3
Non-Learning Curve - Constructed from Chart 5

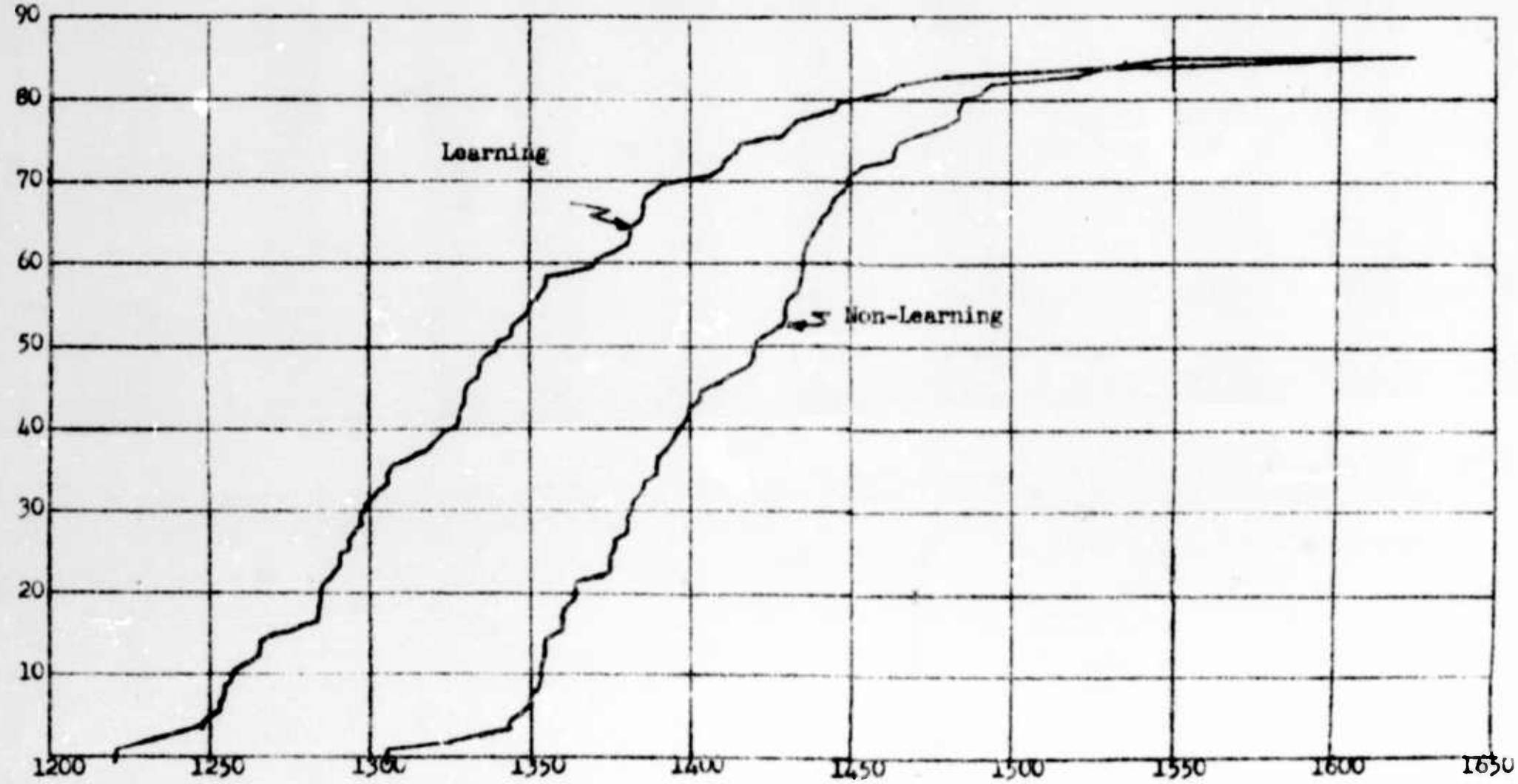


CHART 7

Time Path - Total Schedule Time

10×10×10 Problem

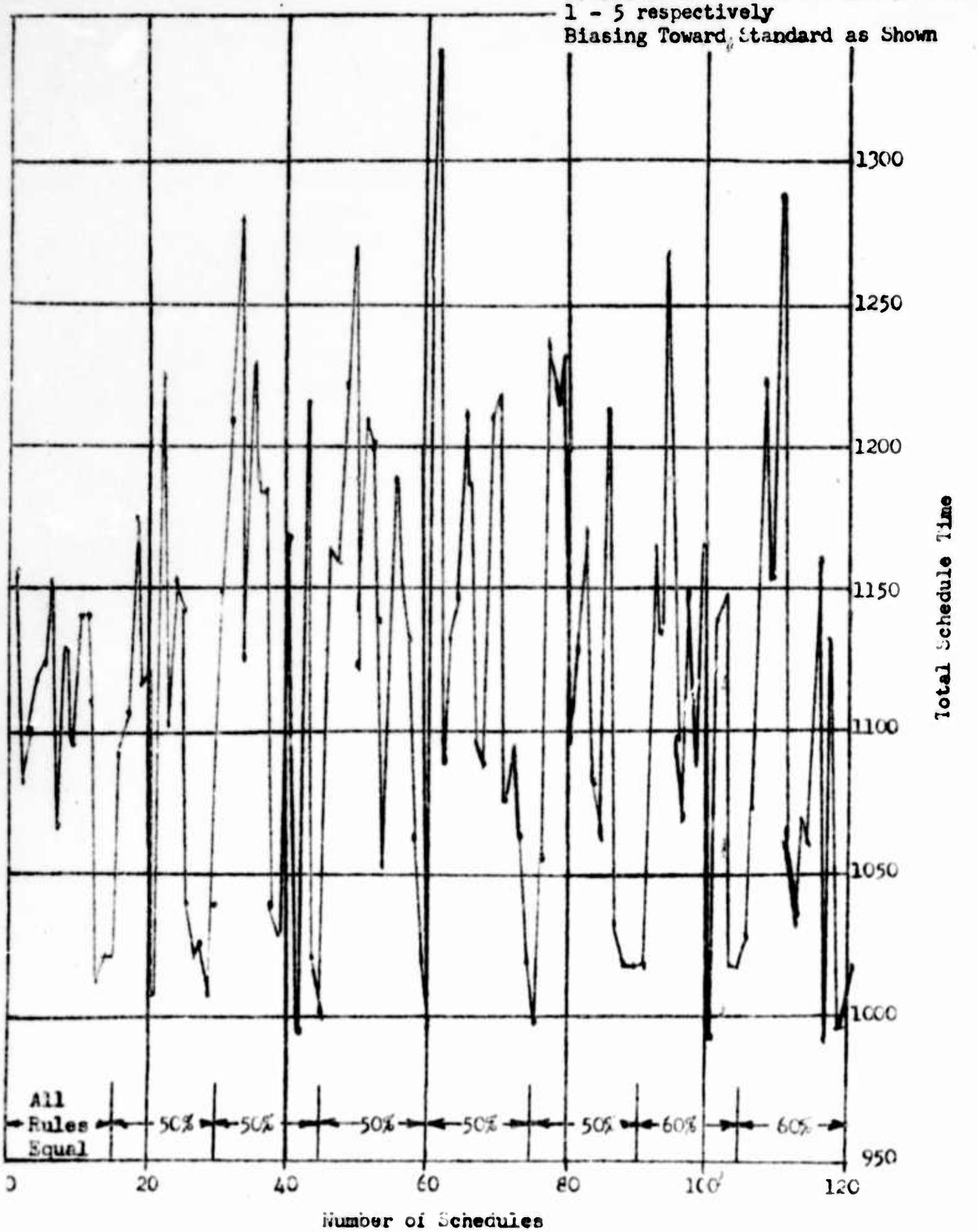


CHART 8

Time Path - Total Schedule Time

10 x 10 x 10 Problem
5 Time Periods - 200 hours each
1,1,5,5,3 Schedules in Time Periods 1-5 Respectively
Biasing Toward standard as Shown

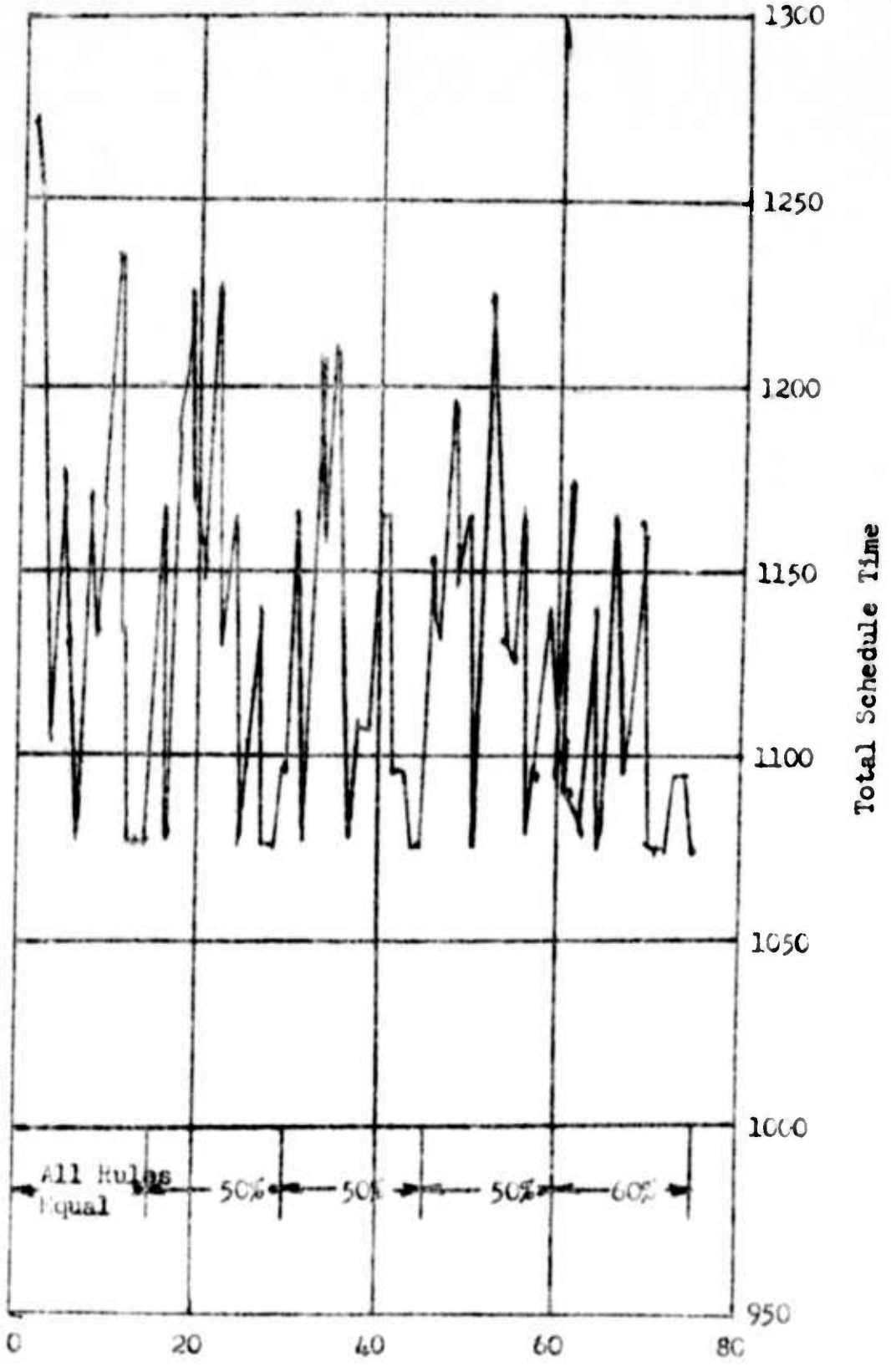


CHART 9

Time Path - Total Schedule Time

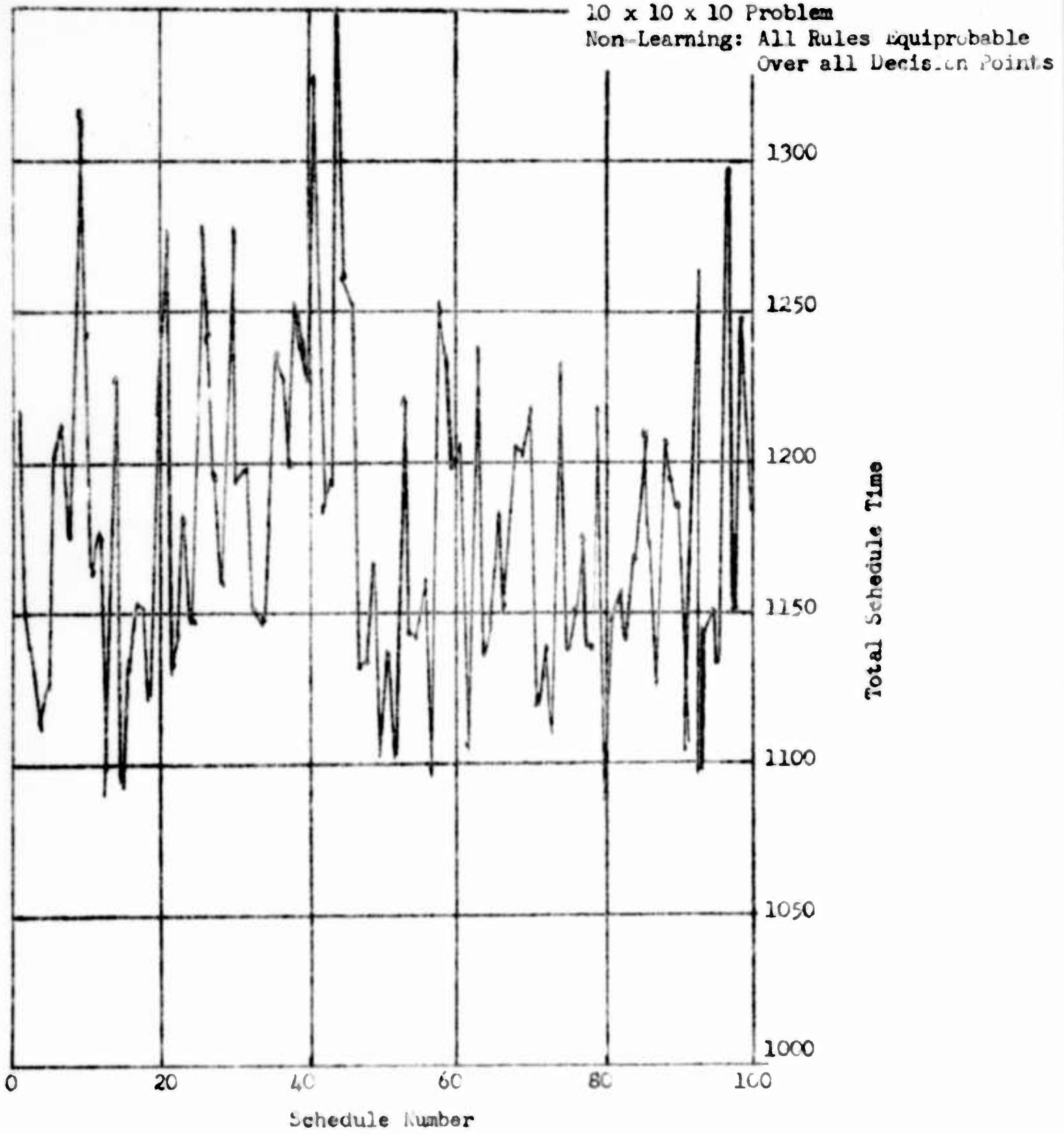


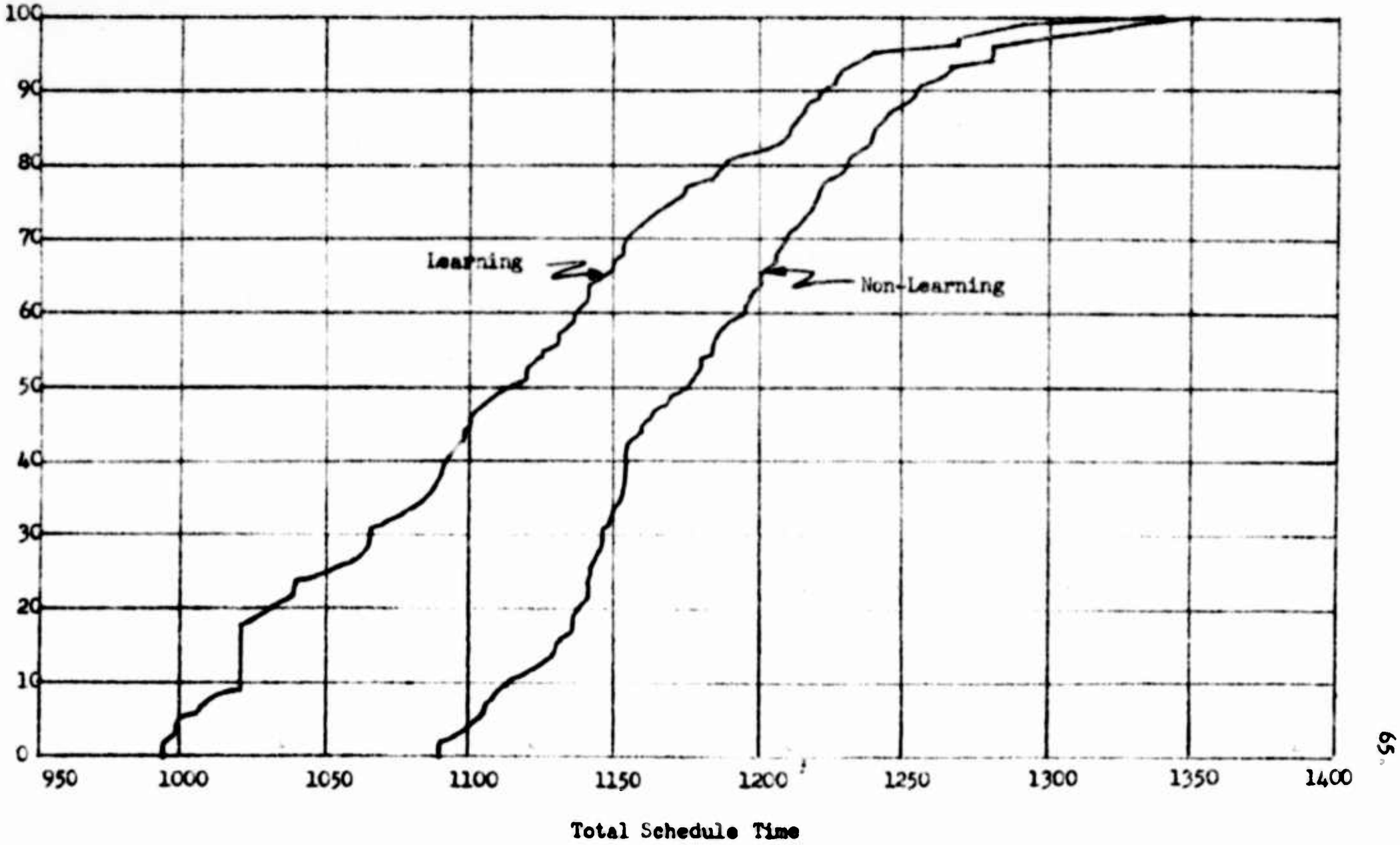
Chart 10

Cumulative Distribution Diagram

10×10×10 Problem

Learning Curve - Constructed from Chart 8

Non-Learning Curve - Constructed from Chart 10



CHAPTER IV

OTHER KINDS OF LEARNING

1. INTRODUCTION

In Chapter II the results of applying simple learning to a series of 6X6 problems was discussed. This simple learning consisted of the application of SIO and LRT decision rules with probabilities varying from SIO:LRT 10:90 to 90:10 in steps of 10. Using the criterion of lowest schedule time the best ratio of SIO:LRT was chosen. The search process then repeated in a narrow zone around the chosen ratio to determine an optimum ratio.

In this chapter, this simple learning is applied to the 10X10 problem. The technique is also extended to multiple learning where varying probabilities of SIO-LRT, SIO-MS and LRT-MS are tested and compared. A further extension uses a variation of the Hooke and Jeeves search routine to explore the effect of combinations of SIO-MS-LRT decision rules. In all instances above, the set of probabilities when chosen remain fixed throughout a complete schedule.

Three general conclusions may be drawn from the studies in this chapter:

(1) For any given length of computing time, simple or multiple learning with local rules will give better results than the same rules applied equiprobably.

(2) When the results of these tests are compared with results of Chapter III, it seems clear that learning by decision point within a schedule is more effective than learning on probabilistic combinations of rules applied over a whole schedule.

(3) The nature of a scheduling problem is changed when the problem is reversed.

2. SIMPLE LEARNING: 10X10 PROBLEM

To decrease the time required for the computation only five combinations of probabilities were tested at each of the two learning stages. In stage one the SIO and LRT decision rules were tested in probabilities varying from SIO-LRT 10:90 to 90:10 in steps of 20 (except where noted). If, for example, the best ratio in stage one was 60:40 SIO:LRT, then in the second stage, ratios from 40:60 to 80:20 in steps of 10 would be tested. The best ratio at this stage would be used in an additional series of tests. The series of tests attempted to discover how few schedules were required for each combination of probabilities at each stage to give a reliable estimate of the best ratio of probabilities. The results are illustrated in Figure 1. As a check on the usefulness of the learning routine, 200 schedules were generated at a ratio of SIO:LRT 50:50. The best time generated was 1049 days.

Discussion of Results: It is clear from the test results that the optimum ratio of SIO:LRT is in the 70:30, 80:20 range. It may be seen that in runs 2, 3, 4 with two schedules generated at each combination the optimum was always found. In runs 5-8, with only one schedule generated at each combination, there was no success in determining the optimum. It would seem that for this problem a minimum of two schedules at each combination is required. A significant result of this test is that all but two of the learning runs produced a better schedule than the series of 200 equiprobable schedules in 15 per cent or less of the time.

Run	Schedules Generated For Each Combinations							
1	10	%SIO Best Time	10 1113	30 1091	50 1098	70 1041	90 1042
	10		50 1063	60 1055	70 1025	80 1008	90 1048
	100					80 1011	
2	2	%SIO Best Time	20 1105	40 1109	60 1096	80 1127	100 1303
	2		40 1120	50 1123	60 1079	70 1053	80 1191
	10					70 1027	
3	2	%SIO Best Time	20 1114	40 1102	60 1122	80 1076	100 1303
	2		60 1119	70 1068	80 1089	90 1096	100 1303
	10			70 1027			
4	2	%SIO Best Time	10 1114	30 1083	50 1115	70 1046	90 1075
	2		50 1082	60 1206	70 1092	80 1124	90 1128
	10					70 1070	

Figure 1

3. MULTIPLE LEARNING

The multiple learning is essentially the same as simple learning. The two stage learning is carried out for combinations of SIO:LRT as described previously. Then this process is repeated for combinations of SIO:MS and of LRT:MS. In these runs it was decided to narrow the range of the second stage so that a best combination of 70:30 in the first stage would be followed by combinations from 60:40 to 80:20 in steps of 5 (except as noted).

This series of tests was conducted for both the 10X10 and the 20X5 problem. The results are shown in Figures 2 and 3 respectively. An attempt was made to determine if the two problems had different characteristics when reversed. Specifically multiple learning was applied to each problem starting at the end of the schedule. The results are also tabulated in the Figures referred to above.

Discussion of results: This series of tests indicates that the best two decision rules for the 10X10 problem are SIO:LRT in approximately the same proportion as found previously (90:10). The best schedule found, 1047 days, happens by chance to be well above the values found in the simple learning. For the 20X5 problem the lowest time, 1252, of this series of tests was found by multiple learning. The best pair of decision rules was clearly found to be SIO:MS. The minimum schedules at all combinations were lower than the majority of the combinations of other rules. For both problems it was clear that more than one schedule must be generated for each combination of probabilities to give reliable results.

The tentative conclusion from the experiment in scheduling the problems when reversed gave lower minimum schedules for most combinations of SIO:MS

SIO:IRT remained the best pair of rules, but the best ratio changed to SIO:LRT 30:70. For the 20X5 problem the SIO:MS combinations became much less effective, while SIO:LRT combinations improved. The best ratio was SIO:LRT 50:50. In both problems the overall minimum for the regular schedule was lower than the overall minimum for the reversed schedule.

Multiple Learning 10X10 Problem

1	SIO-MS	1	%SIO	10	30	50	70	90
			Best Time	1258	1274	1247	1179	1166
		1		50	60	70	80	90
				1177	1155	1110	1185	1221
	SIO-LRT	1	%SIO	10	30	50	70	90
			Best Time	1153	1088	1079	1188	1180
		1		30	40	50	60	70
				1240	1189	1167	1227	1139
	LRT-MS	1	%LRT	10	30	50	70	90
			Best Time	1240	1189	1167	1227	1139
		1		70	80	90	100	
				1138	1275	1240	1177	
	Best Ratio		SIO-LRT 50:50 1079					
2	SIO-MS	1	%SIO	10	30	50	70	90
			Best Time	1246	1246	1208	1163	1203
		1		60	65	70	75	80
				1263	1314	1158	1337	1293
	SIO-LRT	1	%SIO	10	30	50	70	90
			Best Time	1157	1130	1207	1160	1124
		1		80	85	90	95	100
				1126	1090	1166	1161	1275
	LRT-MS	1	%LRT	10	30	50	70	90
			Best Time	1241	1341	1159	1114	1210
		1		60	65	70	75	80
				1252	1275	1203	1178	1191
	Best Ratio		SIO:LRT 85:15 1090					

Figure 2

3	SIO-MS	6	%SIO	10	30	50	70	90
			Best Time	1241	1136	1134	1136	1216
		3		40	45	50	55	60
				1158	1141	1247	1247	1125
	SIO-LRT	6	%SIO	10	30	50	70	90
			Best Time	1115	1048	1070	1077	1047
		3		80	85	90	95	100
				1096	1066	1163	1111	1275
	SIO-MS	6	%LRT	10	30	50	70	90
			Best Time	1270	1107	1114	1113	1097
		3		80	85	90	95	100
				1141	1275	1117	1115	1177
	Best Ratio		SIO:LRT	90:10	1047			
				20X5 Backward				
4	SIO-MS	6	%SIO	10	30	50	70	90
			Best Time	1131	1091	1088	1109	1108
		3		40	45	50	55	60
				1081	1108	1066	1136	1165
	SIO-LRT	6	%SIO	10	30	50	70	90
			Best Time	1077	1062	1082	1083	1109
		3		20	25	30	35	40
				1093	1103	1095	1086	1087
	LRT-MS	6	%LRT	10	30	50	70	90
			Best Time	1106	1090	1065	1068	1101
		3		40	45	50	55	60
				1137	1185	1064	1130	1240
	Best Ratio		SIO:LRT	30:70	1062			

Figure 2

Multiple Learning 20X5 Problem

1	SIO-MS	1	%SIO	10	30	50	70	90
			Best Time	1346	1349	1435	1510	1590
		1		0	5	10	15	20
				1358	1387	1445	1304	1476
	SIO-LRT	1	%SIO	10	30	50	70	90
			Best Time	1556	1594	1607	1537	1437
		1		80	85	90	95	100
				1419	1396	1391	1417	1383
	LRT-MS	1	%LRT	10	30	50	70	90
			Best Time	1403	1555	1532	1534	1544
		1		0	5	10	15	20
				1358	1448	1378	1429	1441
	Best Ratio			SIO:MS	15:85	1304		
2	In Run 2 the best value was obtained from 100% MS 1319							

Figure 3

3	SIO-MS	6	%SIO	10	30	50	70	90
			Best Time	1283	1296	1282	1287	1317
		3		40	45	50	55	60
				1252	1317	1354	1314	1320
	SIO-LRT	6	%SIO	10	30	50	70	90
			Best Time	1416	1496	1460	1346	1361
		3		60	65	70	75	80
				1442	1447	1470	1368	1424
	LRT-MS	6	%LRT	10	30	50	70	90
			Best Time	1282	1365	1421	1448	1448
		3		0	5	10	15	20
				1319	1411	1339	1350	1281
	Best Ratio		SIO-MS	40:60	1252			
				20X5 Backwards				
4	SIO-MS	6	%SIO	10	30	50	70	90
			Best Time	1526	1441	1423	1478	1491
		3		40	45	50	55	60
				1478	1474	1458	1441	1591
	SIO-LRT	6	%SIO	10	30	50	70	90
			Best Time	1374	1348	1276	1288	1290
		3		40	45	50	55	60
				1332	1377	1339	1322	1319
	LRT-MS	6	%LRT	10	30	50	70	90
			Best Time	1522	1427	1429	1427	1425
		3		80	85	90	95	100
				1473	1530	1371	1396	1421
	Best Ratio		SIO-LRT	50:50	1276			

Figure 3

4. DIRECT SEARCH

In order to make the learning process more efficient, a direct search program (see [3]) was written as follows.

1. Generate a schedule using SIO-LRT-MS equiprobable. Use schedule time as first standard.

2. Increase SIO by Δ from base, decreasing LRT, MS by Δ/2. Generate a schedule.

3. Repeat (2) for LRT.

4. Repeat (2) for SIO.

5. Repeat steps 2, 3, 4 N times, or until a schedule is generated below the standard.

6. If the schedule is below the standard, update standard, record the successful rule, go to 8.

7. If Δ = δ go to 10. Otherwise put the value of δ in Δ and go to 2. (δ < Δ)

8. Increase probability of successful rule in base by Δ. Generate a schedule.

9. If the schedule is below standard, update the standard and go to 8: if not, go to (2).

10. Halt.

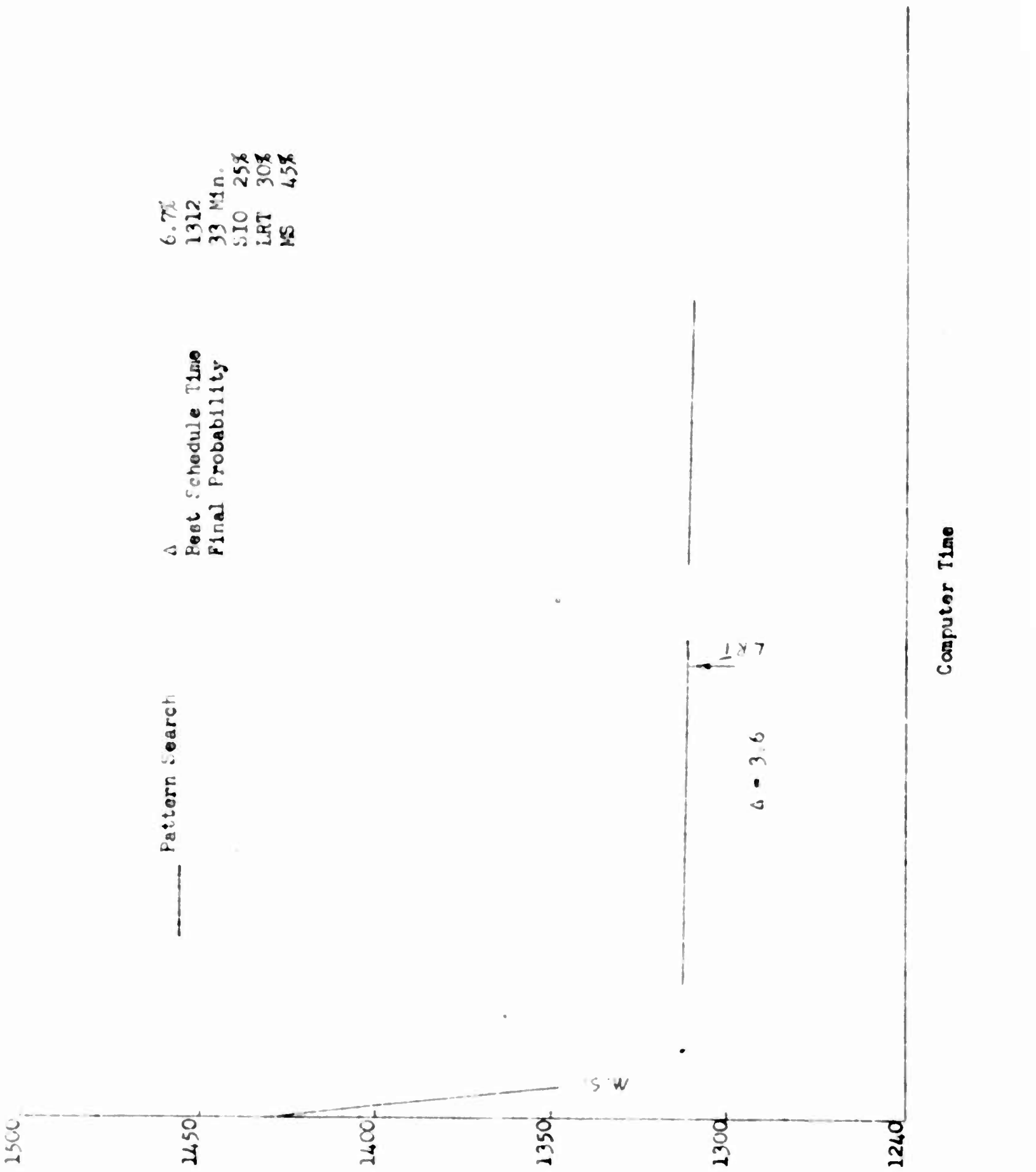
In the series of tests conducted with this program N = 10, Δ was varied from 6.7 to 13.3% and σ was 3.6%. The results for the 20X5 problem are illustrated in Figures 4-6.

Discussion of Results: The graphs show that the technique does give progressively lower times. However, the lowest achieved in all three cases are higher than the time obtained from multiple learning with the same computation time. The tests do show that the higher Δ of 13.3% is more

effective in obtaining schedules and that little benefit is obtained when Δ is reduced to 3.6% for the final ten trials. This suggests that the program could be modified further to make it more efficient.

Figure 4

Pattern Search 20 × 5



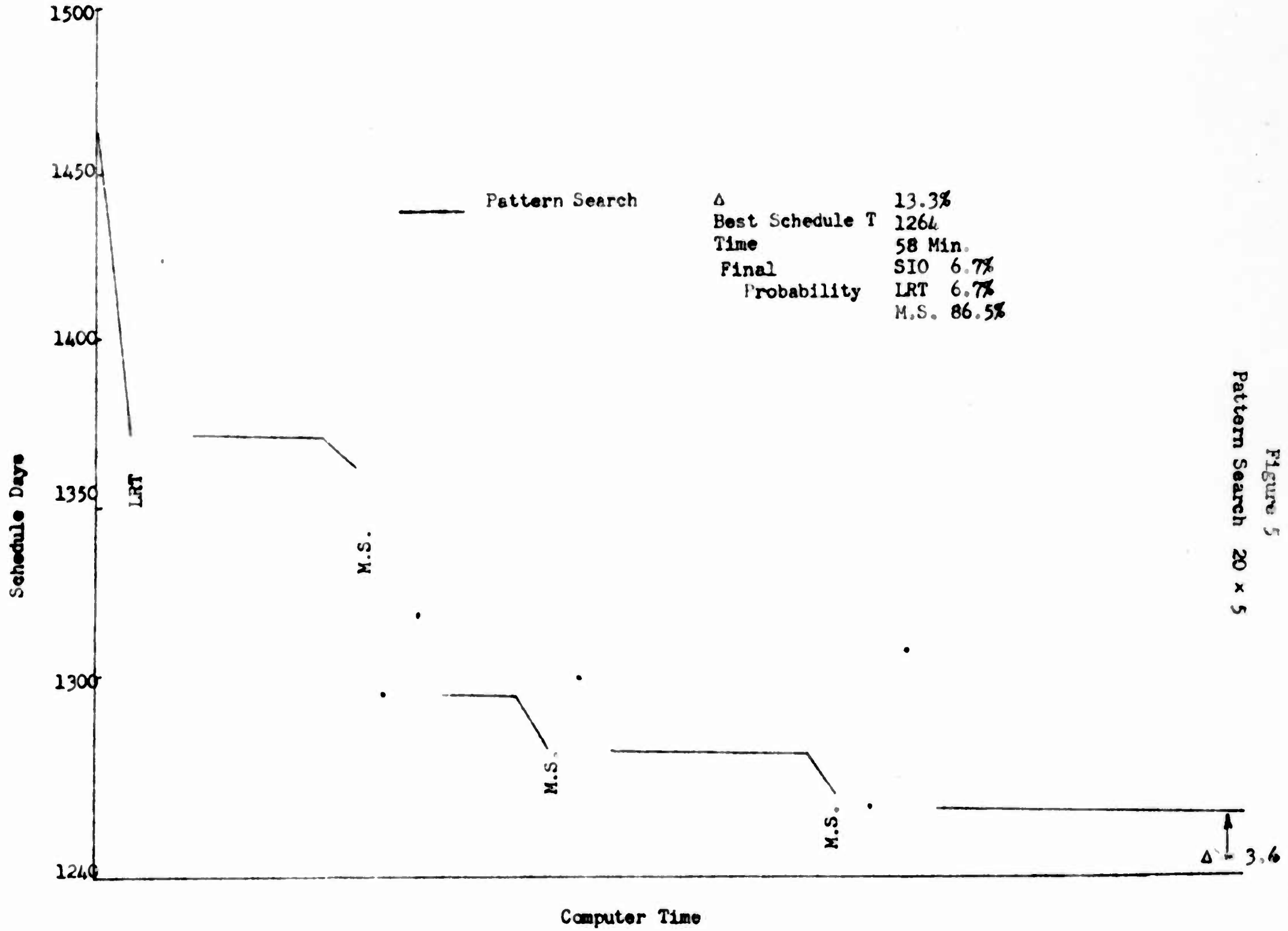
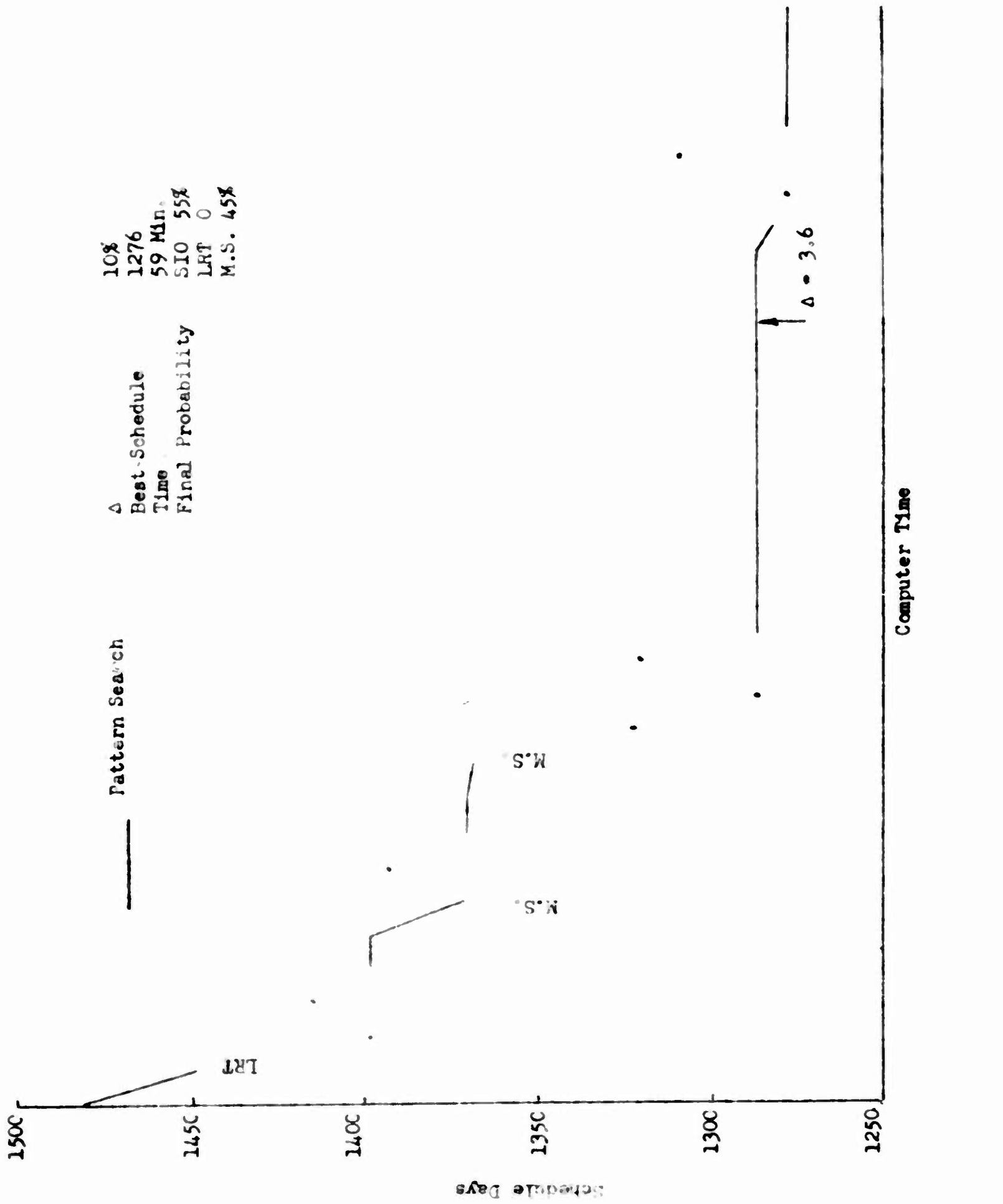


Figure 5
Pattern Search 20 x 5

Figure 6

Pattern Search 20 × 5



CHAPTER V

JOB SHOP SCHEDULING PROBLEM
GOOD INTERCHANGE TECHNIQUE

Problem: Given a feasible schedule, attempt to improve it by interchanging the positions of goods on the schedule of any machine.

Method: The program was written so that scheduling instructions were taken from a specification matrix. Goods would be scheduled on each machine in their order of appearance in this matrix. The original matrix (Figure 1) was generated by the use of the machine slack local scheduling rule. Modifications were made to this matrix in the following manner:

1. A machine was picked at random.
2. Two goods were chosen at random from the schedule of this machine, and their positions on the schedule were interchanged.
3. A schedule was then generated and the finish date recorded.
4. If the finish date was lower than the previous optimum the interchange was retained and a new optimum was set. Otherwise, the specification was returned to its starting position. The process was then repeated a given number of times.

A triple exchange of goods was also investigated. The method was essentially the same as above except that the position of three goods were interchanged.

Results: The results are tabulated in Figure 2. It may be seen that in the series of runs, 2, 3 and 4, the schedule time was reduced from 1334 to 1314 days in 95 minutes of computer time. The triple interchange was not successful in reducing the schedule time in 38 minutes of computer time.

Beginning Specification Matrix

	Machine				
	1	2	3	4	5
Goods	13	11	8	11	11
	20	16	5	5	5
	5	5	13	8	20
	11	8	20	20	8
	15	20	17	9	15
	8	13	19	19	19
	17	19	12	2	16
	9	17	1	16	18
	19	7	11	7	3
	1	15	7	12	1
	2	2	15	10	12
	12	12	10	1	10
	7	1	6	17	3
	16	10	16	13	13
	18	18	18	18	7
	10	6	14	14	4
	6	3	3	3	2
	3	4	2	15	17
	14	14	9	4	14
	4	9	4	6	9

Schedule Time 1334

Figure 1

Pair Interchange

	Cycles	Optimum Schedule			Interchange Goods			Computer Time min.
		Start	Finish	Column				
1	50	1334	1326	4	17	12	-	31 min.
2	50	1334	1322	4	7	5	-	31 min.
3	50	1322	1314	3	18	9		32 min.
4	50	1314	1314	-	-		-	32 min.

Triple Interchange

5	50	1334	1336		-		-	38 min.

Figure 2

CHAPTER VI

PARAMETRIC COMBINATIONS OF LOCAL JOB SHOP RULES

1. INTRODUCTION

The study undertaken in this paper is motivated in part by the success of probabilistic learning combinations of local rules reported on in previous chapters and in part by the impression that a different means of combining local rules could be even more effective. Although the SIO, LRT and MS rules are not directly commensurate, it is possible to change each into a hybrid rule which can then be "parametrically" combined with the others, meaningfully preserving the identity of the components, but in synthesis creating a rule capable of decisions which cannot be specified by any of the rules in isolation. We shall call such a rule a *hybrid rule*.

Empirical results of testing combinations of hybrid local rules appear to confirm these hypotheses, and to uncover some additional advantages of the hybrid rules as well. It should be emphasized that the study undertaken in this paper is of restricted scope. It has not attempted to make use of the apparently strong node-dependency found in the previous chapters, nor has it attempted to combine more than two local rules at once on any given problem. Thus if the combined hybrid rules were superior to the two rule probabilistic combinations with identical weights applied at each decision point, the rationale underlying their creation would be justified. In fact, the best hybrid rule combinations not only surpass the best uniformly weighted probabilistic combinations, but on two of the three problems examined do as well or better than the best probabilistic combination which allows for node-dependency. The best

hybrid combinations were found by generating substantially fewer schedules than required in finding the best probabilistic combinations, and a study of the plot of schedules produced shows that a coarser range of parameter settings would have produced results as good, so that the number of schedules examined was greater than necessary.

2. REQUIREMENTS FOR HYBRID RULE COMBINATIONS

In order to modify a set of simple local rules so that they can be combined, it is necessary to decide on a format for the synthetic rule which they will compose. The format adopted in this paper is well illustrated by the unmodified LRT rule. The LRT rule can be conceived as consisting of two parts: (i) an index function which assigns an index (remaining machining time) to a job based on its current state in an evolving schedule; and (ii) a decision function which selects the job associated with an index having a specified property (the greatest index of those defined). The synthetic rule used here will have the same decision criterion as the LRT rule, that is, it will always select the job associated with the maximum index given by its index function. The simplest way of creating a synthetic rule from a pair of local rules would be simply to sum the indices assigned by each, and apply the decision criterion to the sums. This will not give very sensible results for the SIO and LRT rules, since in the SIO the best index is the smallest, whereas in the LRT the best index is the greatest. Taking the negative of the SIO indices, and then adding them to the LRT indices, would give somewhat more meaningful results. This is still not very reliable, however, since the sizes of the operation times (the SIO indices) may be expected to be similar from one point in the schedule to

the next, whereas the remaining machining times (the LRT indices) are steadily decreasing. The way around this complication is to compute the total number of unperformed operations, divided by the number of jobs, and then to divide this number into the remaining times for the jobs in a machine queue. The outcome gives values for the index function of the hybrid LRT rule which are commensurate with the SIO indices. This can perhaps most easily be seen by noting that if all jobs had the same number of operations remaining, the hybrid LRT indices would give the average remaining operation time for each job in the machine queue. Thus, since at every point in the schedule the hybrid LRT indices belong to the same scale as the imminent operation times of the SIO rule, their combination in a simple algebraic sum (the first plus the negative of the second) has a stable significance.

The synthetic rule, as an extension of the example just given, takes for its index function a linear combination of the index functions of a set of hybrid rules. Thus the difficulty mentioned in summing the SIO and LRT index function values is precisely the one required to be overcome by the hybrid rules which compose the synthetic SIO-LRT combination used in this report. Under the assumption that such difficulties as these have been resolved, the form of the synthetic rule can be made explicit as follows. Let $X_1, X_2, \ldots, X_n$ denote the index functions associated with n local scheduling rules. The index function X for the synthetic rule is given by

$$X = a_1X_1 + a_2X_2 + \ldots + a_nX_n,$$

where the a_j make up an arbitrary set of non-negative real numbers. The index functions are defined so that for a given job S_j, in a particular state in an evolving schedule, $X_i(S_j) = n_{ij}$ is a number. The index

function a_iX_i is characterized in the usual way by $a_iS_i(S_j) = a_in_{ij}$,

and $X(S_j) = a_1n_{1j} + a_2n_{2j} + \cdots + a_nn_{nj}$

If $S = (S_1, S_2, \ldots, S_m)$ is the set of jobs belonging to the queue of a specific machine, then the decision criterion for the synthetic rule may be represented as a function of S producing one of the jobs in S as its value. In this paper $\bar{X}$ is defined so that $\bar{X}(S) = S_k$ where S_k maximizes $X(S_j)$ for all j such that $S_j \in S$.

The requirements to be placed upon the index functions S_i so that they can be combined in a meaningful synthetic rule can now be formulated as follows:

1) The X_i should be defined so that they can be submitted individually to the decision criterion and yield the same decisions as the unmodified local rules from which they are derived.

2) Each X_i must be scaled so that, for any given set of weights (a_i), no index function will alter its relative influence on the composite decision as the schedule evolves.

3) Each X_i in conjunction with the decision criterion must be capable not only of specifying a "best" choice, but must be able to rank all choices meaningfully along a scale which correctly reflects their relative positions. Furthermore, the scale must be linear; that is, its units must be of invariant significance at all points.

4) The X_i must depend only on parameters whose significance is unchanged from problem to problem.

Obviously, the above requirements are the ideal ones. They might be imperfectly met and the synthetic rule still result in excellent schedules

for some values of the a_i. The primary value in satisfying the requirements is the intuitive expectation that uniform changes in the a_i would in consequence be more likely to result in schedules having certain readily specifiable relationships to one another. To express this notion more adequately, let P denote a specific problem, and T_p denote the time interval from the first operation begun to the last completed for that problem. Given a set of X_i and a decision criterion $\bar{X}$, T_p can be represented as a function of the parameters a_i for each schedule generated by the synthetic rule. It is assumed that the synthetic rule always determines a unique choice, so that T_p is single-valued. Then for each setting of the parameters,

$$T_p = T_p (a_1, a_2, \ldots, a_n)$$

denotes the schedule time which we are interested in minimizing.[1/] The restrictions placed upon the X_i are in part an attempt to formulate those conditions which will hopefully make the function T_p the best behaved. The other purpose of the restrictions depends on the assumption that the original local rules were actually relevant to the problem objective, and by defining the hybrid rules properly the degree to which each local rule was relevant could be mirrored by selecting the right values for the a_i, hence allowing an optimal combination.

Once the X_i are given, however, the real concern is with T_p. Empirical investigations to determine the merit of the synthetic rule concept should be aimed at finding answers to questions like the following.

[1/] For more generality it would be possible to specify decision nodes, and a set of a_i associated with each. This essentially multiplies the arguments of the function T_p, but does not affect any other part of the ensuing discussion.

1) Do the lowest values of T_p compare favorably with schedule times produced by other methods?

2) Does T_p as a multivariate function have a unique relative maximum or a unique relative minimum?

3) If not, do the successive relative extrema follow a pattern: for example, are they increasingly high or low?

4) Do the smallest values for T_p occur together, and if so, what range of the parameter settings will yield these values?

5) Do the lowest values of T_p occur frequently - if they do not occur together in a characteristic subspace of parameter settings?

6) Is the synthetic rule consistent for a variety of problems: that is, for each pair of problems p_1 and p_2, is there a constant K (depending on p_1 and p_2) so that

$$K\, T_{p1}(a_1, a_2, \ldots, a_n) \quad T_{p2}(a_1, a_2, \ldots, a_n)$$

for all a_i? Or will the above condition hold with ka_i replacing a_i in the term on the left?

7) Can problems be readily classified for values of the a_i which make T_p small?

3. DESCRIPTION OF TESTS

The three problems examined in the Fisher and Thompson study [1] and in the preceeding chapters of this report were used to test synthetic rules consisting of two hybrid local rules, with the values of a_1 and a_2 held constant in the generation of any single schedule, across all decision points. Problem I consists of six jobs to be routed across six machines, Problem II consists of ten jobs and ten machines, and Problem III consists of twenty jobs and five machines.

Synthetic rules consisting of the hybrid SIO and LRT rules described earlier were applied to all three problems, and in addition a hybrid MS rule was coupled with the LRT rule on Problem III.[1/]

The hybrid MS rule was based on the following assumptions:

1) Once the critical machine - the machine with the greatest processing time yet to be performed - is determined, the relative desirability of scheduling one job instead of another depends only on the difference in times required to reach that machine.

2) The time required by a job until it is available to be processed on the critical machine can be reasonably approximated by the time at which it would be released from the machine for which it is presently queued, plus the sum of operation times to be performed on all machines before the critical one is encountered.

3) If the critical machine is the one whose queue is presently being examined, the MS index function reduces to specifying the times required until the queue jobs will be ready for machining, weighted by a factor of 3.

4) There is no necessity to take into account by how much the critical machine exceeds the second most critical machine in

---

[1/] The hybrid SIO rule actually used defined imminent operation time to be the time required to complete an operation if its associated job were scheduled now, i.e., the time to machine the job operation plus the time consumed, if any, in waiting for the job to arrive. The hybrid SIO index function subtracted the imminent operation time from twice the greatest operation time in the schedule, always producing positive values. Taking the negatives of imminent operation times, as suggested above, would have been both simpler and more in accordance with requirement (2) on page 86, though there would have been no difference in schedules generated.

remaining processing time, nor is it important to consider when queued jobs which reach machines other than the one most critical.

5) When a job does not have any of its remaining operations to be performed on the critical machine, its relative importance by the MS criterion can be approximated by specifying that it reach the critical machine one average operation time after the greatest sum of remaining operation times for any job in the queue.

The foregoing assumptions bear a crude resemblance to reality, but more realistic assumptions would require adjustments of the MS rule which would completely destroy its already borderline local status. Operating within the framework of these assumptions, the hybrid MS index function gives the negatives of the indices produced by the unmodified index function, to accommodate the maximization principle of the synthetic decision criterion.

4. TEST RESULTS

We turn now to a description of the actual handling of the rules and the test results. For each of the three problems on which the rules were tested, the weight of one of the index functions was set equal to unity, and the other allowed to vary from zero to a positive value large enough that the first function did not influence the decision criterion. In each combination, the hybrid LRT rule was the one whose weight was allowed to vary. With each schedule generated the least positive increment (using discrete jumps of .001) of the LRT weight was determined which would cause a different decision to be made at any point in the schedule. In this way it would be possible to generate, exactly once, every unique schedule using non-negative weights for the two hybrid index functions.

(Ties were resolved by picking the first job found which maximized the decision criterion.) By permitting only the LRT weight to vary, the time function T_p was made dependent on a single variable for any given problem and any given set of rules. The plot of T_p for the three problems is given in Figures 1 through 4.

The seventeen schedules generated for Problem I exhaust all of those which can be produced by SIO-LRT combinations using non-negative weights. It is known that there are exactly two best schedules possible with schedule times of 55 hours. The simple enumerations by varying the LRT weight as described above found them both. From Figure 1 it can be seen that any weight for the hybrid LRT lying between 1.61 and 2.5 with the hybrid SIO weight set at unity, would have produced an optimal schedule of 55 hours. Thus, a systematic incrementing of the LRT weight as before, but requiring that no increments be smaller than .89, would have still assured that the best schedule be found, reducing the total number of schedules generated to 4. It is not easy to make a direct comparison with either the probabilistic combination method which uses uniform weights at each decision point (Method A), or the method which allows the weights to vary across decision points (Method B). The reason for this is that the optimal sets of weights for Methods A and B were not determined in a single computer run, but were found in the process of adjusting parameters and techniques of exploration after each run, and then resubmitting the methods to the computer using data obtained from previous exploration attempts. For example, a schedule of 55 hours was found and reproduced on the eleventh through thirteenth of the schedules generated on one computer run with Method A. However,

Problem I (SIO Weight = 1)

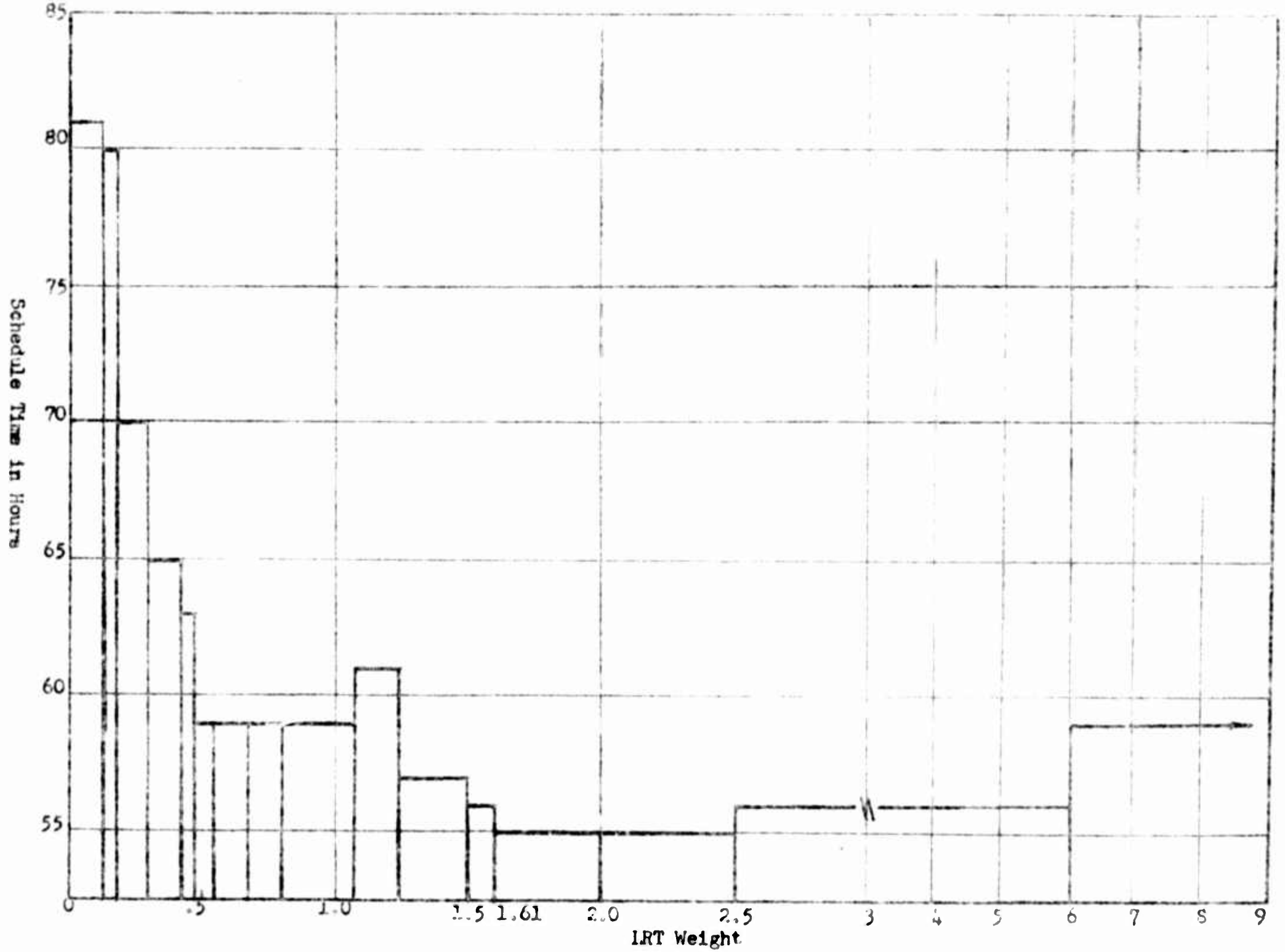


Figure 2

Problem II (SIO Weight = 1)

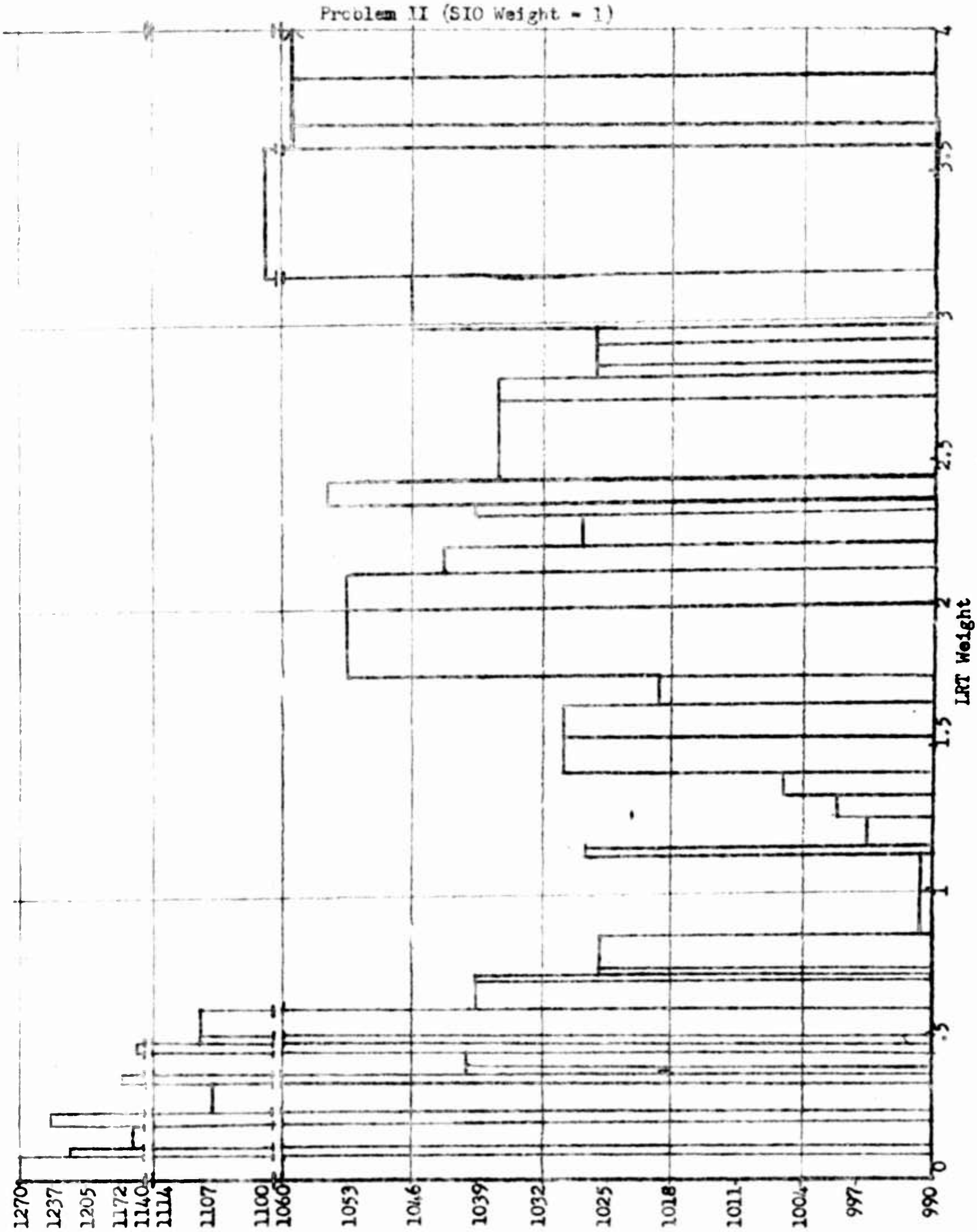
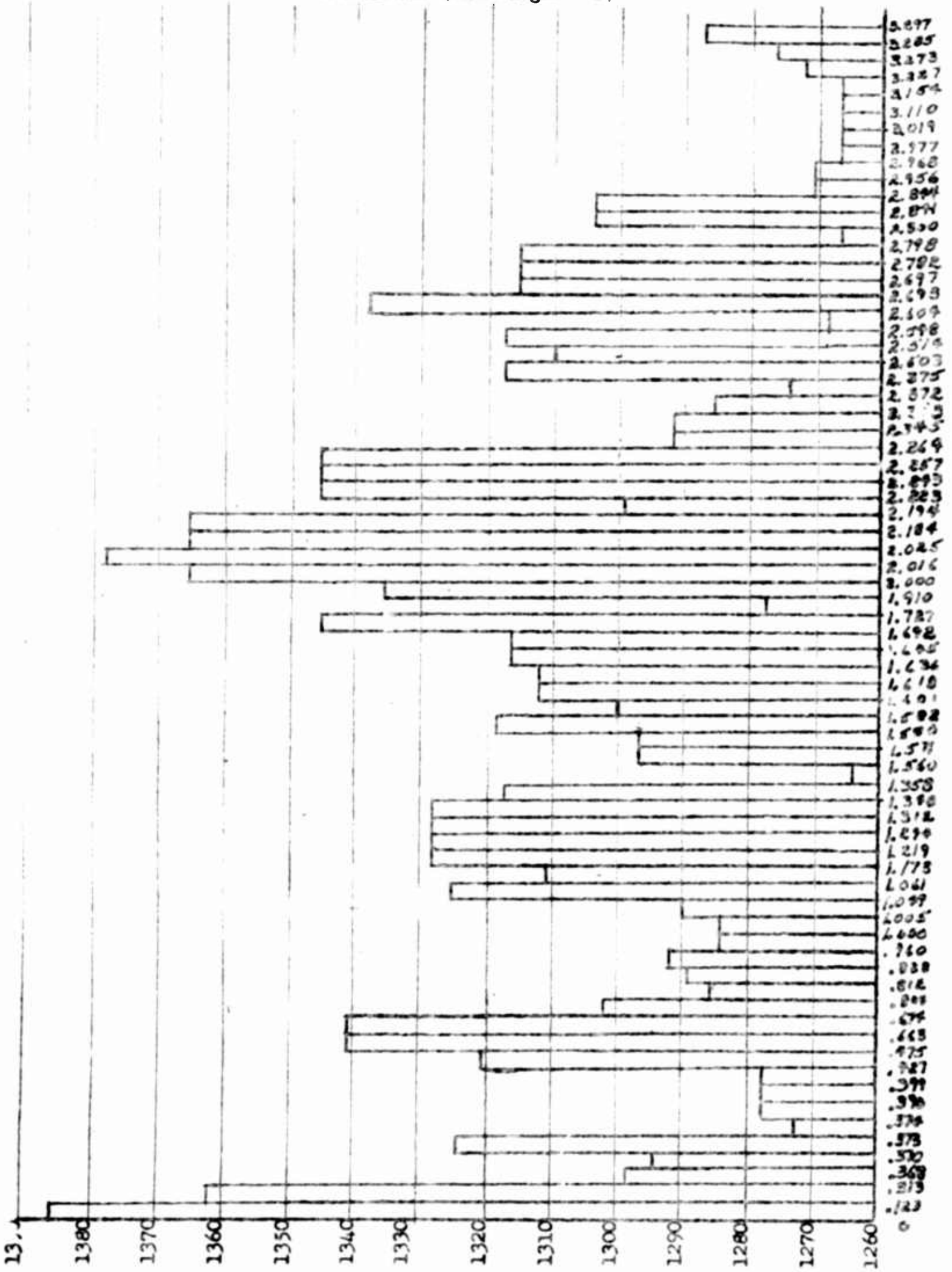


Figure 3

Problem III (SIO Weight - 1)



Problem III

MS - LRT Rule

(MS Weight = 1)

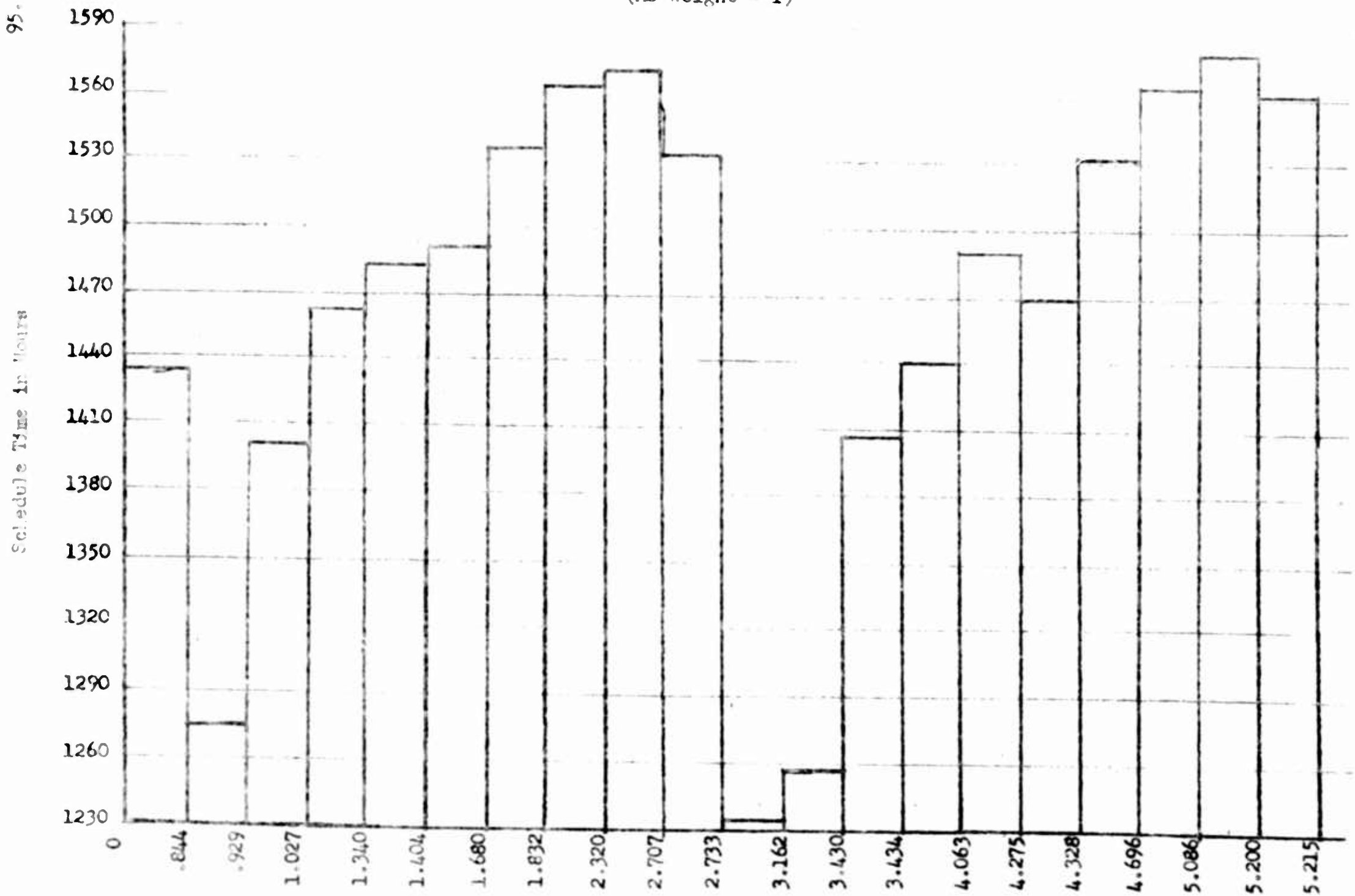


Figure 4

approximately 100 schedules were generated in runs leading up to this achievement. Method B, similarly, on Problem I, produced three or four schedules of 55 hours out of approximately 100 generated. Thus the result obtained by the hybrid rule method of two 55 hour schedules produced in 17 schedules generated indicates that this approach was superior to the probabilistic combination methods on Problem I. The fact that a coarser range of weight settings for the hybrid rule method would have insured finding one 55 hour schedule in only 4 generated strongly supports this conclusion.

For Problem II the hybrid rule approach achieved better schedule times than obtained by either of the probabilistic learning techniques.

The best schedule obtained by Method A had a length of 1008 hours, and the best found with Method B was 994 hours. The last was found once in a computer run which generated 90 schedules. The hybrid SIO-LRT combination located two schedules of 991 hours in 45 schedules generated. Figure 2 shows that a lower bound of increments to the LRT weight of .327 would still have insured that a schedule of 991 hours be produced, and would have cut down the total number of schedules generated to 13.

Problem III is the only problem for which a probabilistic rule was found which did better than any of the hybrid rule combinations located. It should be pointed out, however, that the best schedule was found by Method B, which assigned weights to all three of the SIO, LRT and MS rules in addition to allowing varying weights at different decision points. The time length of this best schedule was 1221 hours, found once in a two-computer-run sequence on the 199th schedule generated out of 240. The best schedule produced by Method A was found using an MS-SIO

combination, and was 1240 hours, found on a computer run producing about 200 schedules. According to available information, 10 schedules were generated by Method A using the best set of weights, only one of which was the optimal. For Problem III SIO-LRT combinations and MS-LRT combinations were tried using the hybrid rules, with a best schedule for the latter at 1238 hours out of 20 schedules generated. While an MS-SIO combination was found to be the best by Method A, there was insufficient time to try combinations of these two rules by the hybrid method. It would be hoped that by permitting the weights to vary at different decision points, the hybrid rule combinations, analogous to the probabilistic combinations, would produce still better schedules. The fact that many more schedules were generated with Method B than by the hybrid rules leaves room to test other combinations for favorable comparison. On the other hand, Problem III presents the hybrid rule concept with a crucial test. As Figure 3 illustrates, the plot of schedule times obtained with SIO-LRT combinations is quite erratic, unlike the better behaved graphs in Figures 1 and 2. If most practical problems produce plots like Problems I and II, then systematic search techniques can be used to find the best weights quickly. But if practical problems are more often structured like Problem III, little more can be done than to generate schedules in much the same manner as done here, picking the best one found. The process still appears considerably more efficient than generating schedules at random, and as already shown the hybrid combinations surpass the results obtained with Method A tried by Crowston, Thompson and Trawick.

5. SUMMARY AND CONCLUSION

Fisher and Thompson showed that it was possible to combine local job shop scheduling rules by a "probabilistic learning" method to create a new local rule superior to any of the individual rules composing it. The new rule consisted of applying one of the basic rules at each decision point, the choice of which basic rule to use made by assigning a weight to each one specifying its probability of selection. The learning concept was introduced to find a good set of weights by examining decision sequences made on past schedules. An extension of the Fisher and Thompson study with modified learning techniques has been made by Crowston, Thompson and Trawick, using probabilistic combinations with weights held uniform across all decision points (Method A), and with weights allowed to vary across the decision points (Method B). Results obtained by both methods have been superior to those obtained in the original study, and Method B has proved superior to Method A. The work in this paper is motivated by the success of the probabilistic learning combinations, and by the feeling that a different means of combining local rules, first changing them into hybrid rules, and then synthesizing these into a rule different from any of its constituents, would be even more successful. The hybrid rule method has the following advantages:

1) It can prescribe choices impossible by an all-or-nothing application of one of the basic rules, as required by the probabilistic combination method.

2) The hybrid rule approach satisfies the intuitive notion that the best combination of a set of rules does not depend on a

mutually exclusive choice among them, but by some means of taking into account the information given by each, establishing a new rule that is more than any one in isolation. Analogously, a marksman who has a tendency to shoot to the left, but whose gun sights are off to the right, will not do best by deciding to compensate only for one or the other of the two defects, but by trying to take both into account at once.

3) As a function of the weights applied to the components of the synthetic rule, the schedule times produced by the hybrid rule method yield a plot which is easier to search for extreme values, and easier to analyze by standard techniques, than the stochastic multi-valued time function associated with the probabilistic combination approach.

4) The hybrid rule method encourages further study to exploit problem structure by its underlying philosophy. Analogously, again, a marksman will learn to make special adjustments on a windy day to maintain his accuracy, whereas one who takes wind velocity into account only a certain portion of the time will tend to overlook its true relation to the other influences on his shooting.

Empirically, the results obtained with the hybrid rule approach are encouraging. The hybrid method was significantly superior to probabilistic learning by Method A on all three of the problems tested, and was better than Method B on two of the three. It is hopefully anticipated that by permitting different weights to be applied at different decision points, and by allowing combinations of more than two rules at a time, the hybrid rule method will be able to produce still better schedules.

On the other hand, there is a possibility, particularly illustrated by Problem III, that universally efficient technique for finding the best set of weights does not exist. This statement is based on the graph of schedule times against the weight settings for the SIO-LRT rule, which yields a figure lacking an easily specifiable trend. This limitation may not be crucial, however, since good schedules appeared frequently throughout the plot, and a small sampling might be expected to produce a schedule by the hybrid method nearly as good as the optimum attainable. Parameter differences of as large as .25 would have assured finding a schedule of 1267 hours on the SIO-LRT combination (it would in fact in this case have found the best at 1264 hours on the sixth schedule generated, but this schedule was produced for a parameter range of only .11). The .25 minimum parameter increment would have reduced the number of schedules generated on Problem III by 4/5, leaving 14 out of 70. Of the 20 schedules investigated using the MS-LRT combination, the best, at 1238 hours, persisted for a range that would have permitted investigating no more than 12. This fact of persistence of good schedules across a relatively large parameter variation, supported even more strongly in Problems I and II, indicates that an effective search technique would be to erect a relatively coarse grid across the range of parameter settings to be investigated. This could be applied economically and effectively even if no other form of regularity was found. It should be noted that the restriction in this paper on the size of the minimum parameter increment has been imposed under the most pessimistic assumption possible - that a schedul time cannot be expected to be found unless it persists

for a parameter range as large as the regions within the grid thrown across the parameter space. In practice, less than 100% probability of finding a best or second best schedule time could still be compatible with a highly successful search strategy.

BIBLIOGRAPHY

[1] Fischer, H., and G. L. Thompson, "Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules," Chapter 15 of Ref. 5.

[2] Giffler, B., and G. L. Thompson, "Algorithms for Solving Production Scheduling Problems," Operations Research 8 (1960) 487-503.

[3] Hooke, R., and T. A. Jeeves, "'Direct Search' Solution of Numerical and Statistical Problems," Jour. of Assoc. for Computing Machinery 8 (1961) 212-229.

[4] Manne, A., "On the Job-Shop Scheduling Problem," Chapter 12 of Ref. 5.

[5] Muth, J. F., and G. L. Thompson, Industrial Scheduling, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1963.

[6] Sherman, G. R., and S. Reiter, "Discrete Optimizing," Inst. for Quant. Research in Economics and Management, Paper No. 37, Purdue University, 1963.